

Description of STM32F3 HAL and low-layer drivers

Introduction

STMCube™ is an STMicroelectronics original initiative to make developers' lives easier by reducing development effort, time and cost. STM32Cube covers the whole STM32 portfolio.

STM32Cube Version 1.x includes:

- STM32CubeMX, a graphical software configuration tool that allows the generation of C initialization code using graphical wizards.
- A comprehensive embedded software platform, delivered per Series (such as STM32CubeF3 for STM32F3 Series).
 - The STM32Cube HAL, STM32 abstraction layer embedded software ensuring maximized portability across the STM32 portfolio
 - Low-layer APIs (LL) offering a fast light-weight expert-oriented layer which is closer to the hardware than the HAL. LL APIs are available only for a set of peripherals.
 - A consistent set of middleware components such as RTOS, USB, TCP/IP, Graphics
 - All embedded software utilities, delivered with a full set of examples.

The HAL driver layer provides a generic multiinstance simple set of APIs (application programming interfaces) to interact with the upper layer (application, libraries and stacks).

The HAL driver APIs are split into two categories: generic APIs which provide common and generic functions for all the STM32 series, and extension APIs which include specific and customized functions for a given line or part number. The HAL drivers include a complete set of ready-to-use APIs which simplify the user application implementation. As an example, the communication peripherals contain APIs to initialize and configure the peripheral, manage data transfers in polling mode, handle interrupts or DMA, and manage communication errors.

The HAL drivers are feature-oriented instead of IP-oriented. As an example, the timer APIs are split into several categories following the functions offered by the IP such as basic timer, capture, or pulse width modulation (PWM). The HAL driver layer implements run-time failure detection by checking the input values of all functions. Such dynamic checking contributes to enhance the firmware robustness. Run-time detection is also suitable for user application development and debugging.

The LL drivers offer hardware services based on the available features of the STM32 peripherals. These services reflect exactly the hardware capabilities and provide atomic operations that must be called following the programming model described in the product line reference manual. As a result, the LL services are not based on standalone processes and do not require any additional memory resources to save their states, counter or data pointers: all operations are performed by changing the associated peripheral registers content. Contrary to the HAL, the LL APIs are not provided for peripherals for which optimized access is not a key feature, or for those requiring heavy software configuration and/or complex upper level stack (such as FSMC or USB).

The HAL and LL drivers are complementary and cover a wide range of applications requirements:

- The HAL offers high-level and feature-oriented APIs, with a high-portability level. They hide the MCU and peripheral complexity to end-user.
- The LL offers low-level APIs at registers level, with better optimization but less portability. They require deep knowledge of the MCU and peripherals specifications.

HAL and LL source code is developed in Strict ANSI-C which makes it independent from the development tools. It is checked with CodeSonar™ static analysis tool. It is fully documented and is MISRA-C 2004 compliant.



Contents

1 Acronyms and definitions.....	28
2 Overview of HAL drivers	30
2.1 HAL and user-application files.....	30
2.1.1 HAL driver files	30
2.1.2 User-application files	31
2.2 HAL data structures	33
2.2.1 Peripheral handle structures	33
2.2.2 Initialization and configuration structure	34
2.2.3 Specific process structures	34
2.3 API classification	34
2.4 Devices supported by HAL drivers	35
2.5 HAL driver rules	39
2.5.1 HAL API naming rules	39
2.5.2 HAL general naming rules	40
2.5.3 HAL interrupt handler and callback functions.....	41
2.6 HAL generic APIs.....	42
2.7 HAL extension APIs	43
2.7.1 HAL extension model overview	43
2.7.2 HAL extension model cases	44
2.8 File inclusion model.....	46
2.9 HAL common resources.....	46
2.10 HAL configuration.....	47
2.11 HAL system peripheral handling	48
2.11.1 Clock.....	48
2.11.2 GPIOs.....	49
2.11.3 Cortex NVIC and SysTick timer.....	50
2.11.4 PWR	51
2.11.5 EXTI.....	51
2.11.6 DMA.....	52
2.12 How to use HAL drivers	54
2.12.1 HAL usage models	54
2.12.2 HAL initialization	55
2.12.3 HAL IO operation process	57
2.12.4 Timeout and error management.....	60
3 Overview of low-layer drivers.....	64

3.1	Low-layer files	64
3.2	Overview of low-layer APIs and naming rules	66
3.2.1	Peripheral initialization functions	66
3.2.2	Peripheral register-level configuration functions	69
4	Cohabiting of HAL and LL	71
4.1	Low-layer driver used in standalone mode.....	71
4.2	Mixed use of low-layer APIs and HAL drivers	71
5	HAL System Driver	72
5.1	HAL Firmware driver API description	72
5.1.1	How to use this driver	72
5.1.2	Initialization and de-initialization functions	72
5.1.3	HAL Control functions.....	72
5.1.4	Detailed description of functions	73
5.2	HAL Firmware driver defines.....	77
5.2.1	HAL.....	77
6	HAL ADC Generic Driver	80
6.1	ADC Firmware driver registers structures	80
6.1.1	__ADC_HandleTypeDef	80
6.2	ADC Firmware driver API description.....	80
6.2.1	ADC peripheral features	80
6.2.2	How to use this driver	81
6.2.3	Initialization and de-initialization functions	85
6.2.4	IO operation functions	85
6.2.5	Peripheral Control functions	86
6.2.6	Peripheral state and errors functions	86
6.2.7	Detailed description of functions	86
6.3	ADC Firmware driver defines	92
6.3.1	ADC	92
7	HAL ADC Extension Driver	95
7.1	ADCEx Firmware driver registers structures	95
7.1.1	ADC_InitTypeDef.....	95
7.1.2	ADC_ChannelConfTypeDef	97
7.1.3	ADC_InjectionConfTypeDef	98
7.1.4	ADC_InjectionConfigTypeDef	100
7.1.5	ADC_AnalogWDGConfTypeDef.....	101
7.1.6	ADC_MultiModeTypeDef.....	101

Contents	UM1786
7.2 ADCEx Firmware driver API description	102
7.2.1 Initialization and de-initialization functions	102
7.2.2 IO operation functions	102
7.2.3 Peripheral Control functions	103
7.2.4 Detailed description of functions	104
7.3 ADCEx Firmware driver defines	110
7.3.1 ADCEx	110
8 HAL CAN Generic Driver.....	124
8.1 CAN Firmware driver registers structures	124
8.1.1 CAN_InitTypeDef.....	124
8.1.2 CAN_FilterConfTypeDef.....	125
8.1.3 CanTxMsgTypeDef.....	126
8.1.4 CanRxMsgTypeDef	126
8.1.5 CAN_HandleTypeDef	127
8.2 CAN Firmware driver API description.....	127
8.2.1 How to use this driver	127
8.2.2 Initialization and de-initialization functions	128
8.2.3 Peripheral State and Error functions	129
8.2.4 Detailed description of functions	129
8.3 CAN Firmware driver defines	132
8.3.1 CAN	132
9 HAL CEC Generic Driver	140
9.1 CEC Firmware driver registers structures	140
9.1.1 CEC_InitTypeDef.....	140
9.1.2 CEC_HandleTypeDef	141
9.2 CEC Firmware driver API description.....	142
9.2.1 How to use this driver	142
9.2.2 Initialization and Configuration functions	142
9.2.3 IO operation functions	142
9.2.4 Peripheral Control function.....	143
9.2.5 Detailed description of functions	143
9.3 CEC Firmware driver defines	146
9.3.1 CEC	146
10 HAL COMP Generic Driver.....	155
10.1 COMP Firmware driver registers structures	155
10.1.1 COMP_InitTypeDef	155
10.1.2 COMP_HandleTypeDef	156

10.2	COMP Firmware driver API description	156
10.2.1	COMP Peripheral features	156
10.2.2	How to use this driver	156
10.2.3	Initialization and de-initialization functions	157
10.2.4	Start Stop operation functions	157
10.2.5	Peripheral Control functions	158
10.2.6	Peripheral State functions	158
10.2.7	Detailed description of functions	158
10.3	COMP Firmware driver defines	161
10.3.1	COMP	161
11	HAL COMP Extension Driver	163
11.1	COMPEx Firmware driver defines	163
11.1.1	COMPEx	163
12	HAL CORTEX Generic Driver	181
12.1	CORTEX Firmware driver registers structures	181
12.1.1	MPU_Region_InitTypeDef	181
12.2	CORTEX Firmware driver API description	182
12.2.1	How to use this driver	182
12.2.2	Initialization and de-initialization functions	182
12.2.3	Peripheral Control functions	183
12.2.4	Detailed description of functions	183
12.3	CORTEX Firmware driver defines	188
12.3.1	CORTEX	188
13	HAL CRC Generic Driver	191
13.1	CRC Firmware driver registers structures	191
13.1.1	CRC_InitTypeDef	191
13.1.2	CRC_HandleTypeDef	192
13.2	CRC Firmware driver API description	192
13.2.1	How to use this driver	192
13.2.2	Initialization and de-initialization functions	193
13.2.3	Peripheral Control functions	193
13.2.4	Peripheral State functions	193
13.2.5	Detailed description of functions	193
13.3	CRC Firmware driver defines	195
13.3.1	CRC	195
14	HAL CRC Extension Driver	198

Contents	UM1786
14.1 CRCEEx Firmware driver API description	198
14.1.1 How to use this driver	198
14.1.2 Detailed description of functions	198
14.2 CRCEEx Firmware driver defines.....	199
14.2.1 CRCEx.....	199
15 HAL DAC Generic Driver.....	201
15.1 DAC Firmware driver registers structures	201
15.1.1 DAC_ChannelConfTypeDef	201
15.1.2 __DAC_HandleTypeDef	201
15.2 DAC Firmware driver API description.....	202
15.2.1 DAC Peripheral features.....	202
15.2.2 How to use this driver	203
15.2.3 Initialization and de-initialization functions	204
15.2.4 IO operation functions	204
15.2.5 Peripheral Control functions	205
15.2.6 DAC Peripheral State and Error functions.....	205
15.2.7 Detailed description of functions	205
15.3 DAC Firmware driver defines	210
15.3.1 DAC	210
16 HAL DAC Extension Driver	215
16.1 DACEEx Firmware driver API description	215
16.1.1 How to use this driver	215
16.1.2 Peripheral Control functions	215
16.1.3 IO operation functions	215
16.1.4 Detailed description of functions	216
16.2 DACEEx Firmware driver defines	219
16.2.1 DACEEx	219
17 HAL DMA Generic Driver	220
17.1 DMA Firmware driver registers structures	220
17.1.1 DMA_InitTypeDef	220
17.1.2 __DMA_HandleTypeDef.....	220
17.2 DMA Firmware driver API description	221
17.2.1 How to use this driver	221
17.2.2 Initialization and de-initialization functions	222
17.2.3 IO operation functions	222
17.2.4 State and Errors functions	223
17.2.5 Detailed description of functions	223

17.3	DMA Firmware driver defines.....	226
17.3.1	DMA.....	226
18	HAL DMA Extension Driver.....	231
18.1	DMAEx Firmware driver defines.....	231
18.1.1	DMAEx.....	231
19	HAL FLASH Generic Driver.....	233
19.1	FLASH Firmware driver registers structures	233
19.1.1	FLASH_ProcessTypeDef	233
19.2	FLASH Firmware driver API description.....	233
19.2.1	FLASH peripheral features	233
19.2.2	How to use this driver	234
19.2.3	Peripheral Control functions	234
19.2.4	Peripheral Errors functions	234
19.2.5	Detailed description of functions	235
19.3	FLASH Firmware driver defines	237
19.3.1	FLASH	237
20	HAL FLASH Extension Driver	241
20.1	FLASHEx Firmware driver registers structures	241
20.1.1	FLASH_EraselInitTypeDef	241
20.1.2	FLASH_OBProgramInitTypeDef	241
20.2	FLASHEx Firmware driver API description.....	242
20.2.1	FLASH Erasing Programming functions.....	242
20.2.2	Option Bytes Programming functions.....	242
20.2.3	Detailed description of functions	242
20.3	FLASHEx Firmware driver defines	244
20.3.1	FLASHEx	244
21	HAL GPIO Generic Driver.....	247
21.1	GPIO Firmware driver registers structures	247
21.1.1	GPIO_InitTypeDef	247
21.2	GPIO Firmware driver API description	247
21.2.1	GPIO Peripheral features	247
21.2.2	How to use this driver	248
21.2.3	Initialization and de-initialization functions	248
21.2.4	IO operation functions	248
21.2.5	Detailed description of functions	249
21.3	GPIO Firmware driver defines.....	251

21.3.1	GPIO	251
22	HAL GPIO Extension Driver	254
22.1	GPIOEx Firmware driver defines	254
22.1.1	GPIOEx	254
23	HAL HRTIM Generic Driver	257
23.1	HRTIM Firmware driver registers structures	257
23.1.1	HRTIM_InitTypeDef	257
23.1.2	HRTIM_TimerParamTypeDef	257
23.1.3	_HRTIM_HandleTypeDef	258
23.1.4	HRTIM_TimeBaseCfgTypeDef	259
23.1.5	HRTIM_SimpleOCChannelCfgTypeDef	259
23.1.6	HRTIM_SimplePWMChannelCfgTypeDef	259
23.1.7	HRTIM_SimpleCaptureChannelCfgTypeDef	260
23.1.8	HRTIM_SimpleOnePulseChannelCfgTypeDef	260
23.1.9	HRTIM_TimerCfgTypeDef	261
23.1.10	HRTIM_CompareCfgTypeDef	263
23.1.11	HRTIM_CaptureCfgTypeDef	263
23.1.12	HRTIM_OutputCfgTypeDef	263
23.1.13	HRTIM_TimerEventFilteringCfgTypeDef	264
23.1.14	HRTIM_DeadTimeCfgTypeDef	264
23.1.15	HRTIM_ChopperModeCfgTypeDef	265
23.1.16	HRTIM_EventCfgTypeDef	265
23.1.17	HRTIM_FaultCfgTypeDef	266
23.1.18	HRTIM_BurstModeCfgTypeDef	266
23.1.19	HRTIM_ADCTriggerCfgTypeDef	267
23.2	HRTIM Firmware driver API description	267
23.2.1	Simple mode v.s. waveform mode	267
23.2.2	How to use this driver	268
23.2.3	Initialization and Time Base Configuration functions	271
23.2.4	Simple time base mode functions	272
23.2.5	Simple output compare functions	272
23.2.6	Simple PWM output functions	272
23.2.7	Simple input capture functions	273
23.2.8	Simple one pulse functions	273
23.2.9	HRTIM configuration functions	274
23.2.10	HRTIM timer configuration and control functions	274
23.2.11	Peripheral State functions	275
23.2.12	Detailed description of functions	276

23.3	HRTIM Firmware driver defines	321
23.3.1	HRTIM	321
24	HAL I2C Generic Driver	362
24.1	I2C Firmware driver registers structures	362
24.1.1	I2C_InitTypeDef.....	362
24.1.2	__I2C_HandleTypeDef.....	362
24.2	I2C Firmware driver API description.....	363
24.2.1	How to use this driver	363
24.2.2	Initialization and de-initialization functions	368
24.2.3	IO operation functions	368
24.2.4	Peripheral State, Mode and Error functions	370
24.2.5	Detailed description of functions	370
24.3	I2C Firmware driver defines	382
24.3.1	I2C	382
25	HAL I2C Extension Driver	389
25.1	I2CEEx Firmware driver API description	389
25.1.1	I2C peripheral Extended features.....	389
25.1.2	How to use this driver	389
25.1.3	Extended features functions	389
25.1.4	Detailed description of functions	389
25.2	I2CEEx Firmware driver defines	391
25.2.1	I2CEEx	391
26	HAL I2S Generic Driver	392
26.1	I2S Firmware driver registers structures	392
26.1.1	I2S_InitTypeDef.....	392
26.1.2	I2S_HandleTypeDef	392
26.2	I2S Firmware driver API description.....	393
26.2.1	How to use this driver	393
26.2.2	Initialization and de-initialization functions	395
26.2.3	IO operation functions	395
26.2.4	Peripheral State and Errors functions	396
26.2.5	Detailed description of functions	397
26.3	I2S Firmware driver defines	402
26.3.1	I2S	402
27	HAL I2S Extension Driver	407
27.1	I2SEEx Firmware driver API description.....	407

Contents	UM1786
27.1.1 I2S Extended features	407
27.1.2 How to use this driver	407
27.1.3 Extended features Functions.....	408
27.1.4 Detailed description of functions	408
27.2 I2SEEx Firmware driver defines	411
27.2.1 I2SEEx	411
28 HAL IRDA Generic Driver.....	413
28.1 IRDA Firmware driver registers structures	413
28.1.1 IRDA_InitTypeDef.....	413
28.1.2 IRDA_HandleTypeDef	413
28.2 IRDA Firmware driver API description.....	414
28.2.1 How to use this driver	414
28.2.2 Initialization and Configuration functions.....	416
28.2.3 IO operation functions	416
28.2.4 Peripheral State and Error functions	418
28.2.5 Detailed description of functions	418
28.3 IRDA Firmware driver defines	426
28.3.1 IRDA	426
29 HAL IRDA Extension Driver	435
29.1 IRDAEEx Firmware driver defines	435
29.1.1 IRDAEEx	435
30 HAL IWDG Generic Driver.....	436
30.1 IWDG Firmware driver registers structures	436
30.1.1 IWDG_InitTypeDef	436
30.1.2 IWDG_HandleTypeDef.....	436
30.2 IWDG Firmware driver API description	436
30.2.1 IWDG Generic features	436
30.2.2 How to use this driver	437
30.2.3 Initialization and Start functions.....	437
30.2.4 IO operation functions	437
30.2.5 Detailed description of functions	438
30.3 IWDG Firmware driver defines	438
30.3.1 IWDG	438
31 HAL NAND Generic Driver	440
31.1 NAND Firmware driver registers structures.....	440
31.1.1 NAND_IDTypeDef	440

31.1.2	NAND_AddressTypeDef.....	440
31.1.3	NAND_DeviceConfigTypeDef	440
31.1.4	NAND_HandleTypeDef	441
31.2	NAND Firmware driver API description	441
31.2.1	How to use this driver	441
31.2.2	NAND Initialization and de-initialization functions	442
31.2.3	NAND Input and Output functions	442
31.2.4	NAND Control functions	443
31.2.5	NAND State functions.....	443
31.2.6	Detailed description of functions	443
31.3	NAND Firmware driver defines.....	449
31.3.1	NAND.....	449
32	HAL NOR Generic Driver.....	450
32.1	NOR Firmware driver registers structures	450
32.1.1	NOR_IDTypeDef	450
32.1.2	NOR_CFITypeDef	450
32.1.3	NOR_HandleTypeDef.....	450
32.2	NOR Firmware driver API description	451
32.2.1	How to use this driver	451
32.2.2	NOR Initialization and de_initialization functions	451
32.2.3	NOR Input and Output functions	452
32.2.4	NOR Control functions.....	452
32.2.5	NOR State functions.....	452
32.2.6	Detailed description of functions	452
32.3	NOR Firmware driver defines.....	457
32.3.1	NOR.....	457
33	HAL OPAMP Generic Driver	458
33.1	OPAMP Firmware driver registers structures	458
33.1.1	OPAMP_InitTypeDef	458
33.1.2	OPAMP_HandleTypeDef.....	459
33.2	OPAMP Firmware driver API description	459
33.2.1	OPAMP Peripheral Features	459
33.2.2	How to use this driver	460
33.2.3	Initialization and de-initialization functions	461
33.2.4	IO operation functions	461
33.2.5	Peripheral Control functions	461
33.2.6	Peripheral State functions	461

Contents	UM1786
33.2.7 Detailed description of functions	462
33.3 OPAMP Firmware driver defines.....	464
33.3.1 OPAMP	464
34 HAL OPAMP Extension Driver.....	467
34.1 OPAMPEx Firmware driver API description	467
34.1.1 Detailed description of functions	467
35 HAL PCCARD Generic Driver	468
35.1 PCCARD Firmware driver registers structures.....	468
35.1.1 PCCARD_HandleTypeDef	468
35.2 PCCARD Firmware driver API description	468
35.2.1 How to use this driver	468
35.2.2 PCCARD Initialization and de-initialization functions	469
35.2.3 PCCARD Input Output and memory functions	469
35.2.4 PCCARD Peripheral State functions	469
35.2.5 Detailed description of functions	469
35.3 PCCARD Firmware driver defines.....	473
35.3.1 PCCARD	473
36 HAL PCD Generic Driver	474
36.1 PCD Firmware driver registers structures	474
36.1.1 PCD_InitTypeDef.....	474
36.1.2 PCD_EPTTypeDef.....	474
36.1.3 PCD_HandleTypeDef	475
36.2 PCD Firmware driver API description.....	476
36.2.1 How to use this driver	476
36.2.2 Initialization and de-initialization functions	476
36.2.3 IO operation functions	476
36.2.4 Peripheral Control functions	477
36.2.5 Peripheral State functions	477
36.2.6 Detailed description of functions	477
36.3 PCD Firmware driver defines	484
36.3.1 PCD	484
37 HAL PCD Extension Driver	486
37.1 PCDEx Firmware driver API description	486
37.1.1 Extended Peripheral Control functions.....	486
37.1.2 Detailed description of functions	486
37.2 PCDEx Firmware driver defines	487

37.2.1	PCDEx.....	487
38	HAL PWR Generic Driver	488
38.1	PWR Firmware driver API description.....	488
38.1.1	Initialization and de-initialization functions	488
38.1.2	Peripheral Control functions	488
38.1.3	Detailed description of functions	490
38.2	PWR Firmware driver defines	493
38.2.1	PWR	493
39	HAL PWR Extension Driver	495
39.1	PWREx Firmware driver registers structures	495
39.1.1	PWR_PVDTTypeDef	495
39.2	PWREx Firmware driver API description.....	495
39.2.1	Peripheral Extended control functions.....	495
39.2.2	Detailed description of functions	496
39.3	PWREx Firmware driver defines	497
39.3.1	PWREx	497
40	HAL RCC Generic Driver.....	500
40.1	RCC Firmware driver registers structures	500
40.1.1	RCC_PLLInitTypeDef	500
40.1.2	RCC_OscInitTypeDef	500
40.1.3	RCC_ClkInitTypeDef	501
40.2	RCC Firmware driver API description	501
40.2.1	RCC specific features	501
40.2.2	RCC Limitations.....	502
40.2.3	Initialization and de-initialization functions	502
40.2.4	Peripheral Control functions	503
40.2.5	Detailed description of functions	503
40.3	RCC Firmware driver defines	507
40.3.1	RCC	507
41	HAL RCC Extension Driver	526
41.1	RCCEx Firmware driver registers structures	526
41.1.1	RCC_PерiphCLKInitTypeDef	526
41.2	RCCEx Firmware driver API description	527
41.2.1	Extended Peripheral Control functions	527
41.2.2	Detailed description of functions	527
41.3	RCCEx Firmware driver defines.....	528

41.3.1	RCCEx	528
42	HAL RTC Generic Driver	544
42.1	RTC Firmware driver registers structures	544
42.1.1	RTC_InitTypeDef.....	544
42.1.2	RTC_TimeTypeDef.....	544
42.1.3	RTC_DateTypeDef	545
42.1.4	RTC_AlarmTypeDef	545
42.1.5	RTC_HandleTypeDef	546
42.2	RTC Firmware driver API description.....	546
42.2.1	RTC Operating Condition	546
42.2.2	Backup Domain Reset.....	547
42.2.3	Backup Domain Access.....	547
42.2.4	How to use RTC Driver.....	547
42.2.5	RTC and low power modes	548
42.2.6	Initialization and de-initialization functions	548
42.2.7	RTC Time and Date functions	549
42.2.8	RTC Alarm functions	549
42.2.9	Detailed description of functions	549
42.3	RTC Firmware driver defines	554
42.3.1	RTC	554
43	HAL RTC Extension Driver	564
43.1	RTCEx Firmware driver registers structures	564
43.1.1	RTC_TamperTypeDef	564
43.2	RTCEx Firmware driver API description.....	564
43.2.1	How to use this driver	564
43.2.2	RTC TimeStamp and Tamper functions	565
43.2.3	RTC Wake-up functions	565
43.2.4	Extended Peripheral Control functions	566
43.2.5	Extended features functions	566
43.2.6	Detailed description of functions	566
43.3	RTCEx Firmware driver defines	575
43.3.1	RTCEx	575
44	HAL SDADC Generic Driver	592
44.1	SDADC Firmware driver registers structures	592
44.1.1	SDADC_InitTypeDef.....	592
44.1.2	SDADC_HandleTypeDef	592
44.1.3	SDADC_ConfParamTypeDef	593

44.2	SDADC Firmware driver API description	593
44.2.1	SDADC specific features	593
44.2.2	How to use this driver	594
44.2.3	Initialization and de-initialization functions	596
44.2.4	Peripheral control functions	596
44.2.5	IO operation functions	596
44.2.6	ADC Peripheral State functions.....	598
44.2.7	Detailed description of functions	598
44.3	SDADC Firmware driver defines	610
44.3.1	SDADC	610
45	HAL SMARTCARD Generic Driver.....	616
45.1	SMARTCARD Firmware driver registers structures	616
45.1.1	SMARTCARD_InitTypeDef	616
45.1.2	SMARTCARD_AdvFeatureInitTypeDef.....	617
45.1.3	SMARTCARD_HandleTypeDef.....	618
45.2	SMARTCARD Firmware driver API description.....	619
45.2.1	How to use this driver	619
45.2.2	Initialization and Configuration functions	621
45.2.3	IO operation functions	621
45.2.4	Peripheral State and Errors functions	623
45.2.5	Detailed description of functions	623
45.3	SMARTCARD Firmware driver defines	631
45.3.1	SMARTCARD.....	631
46	HAL SMARTCARD Extension Driver.....	642
46.1	SMARTCARDEX Firmware driver API description	642
46.1.1	SMARTCARD peripheral extended features.....	642
46.1.2	Peripheral Control functions	642
46.1.3	Detailed description of functions	642
47	HAL SMBUS Generic Driver.....	644
47.1	SMBUS Firmware driver registers structures	644
47.1.1	SMBUS_InitTypeDef	644
47.1.2	SMBUS_HandleTypeDef.....	645
47.2	SMBUS Firmware driver API description	646
47.2.1	How to use this driver	646
47.2.2	Initialization and de-initialization functions	647
47.2.3	IO operation functions	648

Contents	UM1786
47.2.4 Peripheral State and Errors functions	649
47.2.5 Detailed description of functions	649
47.3 SMBUS Firmware driver defines	656
47.3.1 SMBUS	656
48 HAL SPI Generic Driver.....	664
48.1 SPI Firmware driver registers structures	664
48.1.1 SPI_InitTypeDef	664
48.1.2 __SPI_HandleTypeDef.....	665
48.2 SPI Firmware driver API description	666
48.2.1 How to use this driver	666
48.2.2 Initialization and de-initialization functions	667
48.2.3 IO operation functions	667
48.2.4 Peripheral State and Errors functions	668
48.2.5 Detailed description of functions	668
48.3 SPI Firmware driver defines	675
48.3.1 SPI	675
49 HAL SPI Extension Driver.....	682
49.1 SPIEx Firmware driver API description	682
49.1.1 IO operation functions	682
49.1.2 Detailed description of functions	682
50 HAL SRAM Generic Driver.....	683
50.1 SRAM Firmware driver registers structures.....	683
50.1.1 SRAM_HandleTypeDef	683
50.2 SRAM Firmware driver API description	683
50.2.1 How to use this driver	683
50.2.2 SRAM Initialization and de_initialization functions	684
50.2.3 SRAM Input and Output functions	684
50.2.4 SRAM Control functions	684
50.2.5 SRAM State functions	685
50.2.6 Detailed description of functions	685
50.3 SRAM Firmware driver defines	689
50.3.1 SRAM	689
51 HAL TIM Generic Driver	690
51.1 TIM Firmware driver registers structures.....	690
51.1.1 TIM_Base_InitTypeDef.....	690
51.1.2 TIM_OC_InitTypeDef.....	690

51.1.3	TIM_OnePulse_InitTypeDef	691
51.1.4	TIM_IC_InitTypeDef	692
51.1.5	TIM_Encoder_InitTypeDef	692
51.1.6	TIM_ClockConfigTypeDef	693
51.1.7	TIM_ClearInputConfigTypeDef.....	693
51.1.8	TIM_SlaveConfigTypeDef	694
51.1.9	TIM_HandleTypeDef	694
51.2	TIM Firmware driver API description	695
51.2.1	TIMER Generic features.....	695
51.2.2	How to use this driver	695
51.2.3	Time Base functions	696
51.2.4	Time Output Compare functions	696
51.2.5	Time PWM functions	697
51.2.6	Time Input Capture functions	697
51.2.7	Time One Pulse functions	698
51.2.8	Time Encoder functions.....	698
51.2.9	IRQ handler management	699
51.2.10	Peripheral Control functions	699
51.2.11	TIM Callbacks functions	699
51.2.12	Peripheral State functions	700
51.2.13	Detailed description of functions	700
51.3	TIM Firmware driver defines.....	728
51.3.1	TIM.....	728
52	HAL TIM Extension Driver.....	746
52.1	TIMEx Firmware driver registers structures.....	746
52.1.1	TIM_HallSensor_InitTypeDef	746
52.1.2	TIM_BreakDeadTimeConfigTypeDef	746
52.1.3	TIM_MasterConfigTypeDef	747
52.2	TIMEx Firmware driver API description	747
52.2.1	TIMER Extended features	747
52.2.2	How to use this driver	748
52.2.3	Timer Hall Sensor functions	748
52.2.4	Timer Complementary Output Compare functions.....	749
52.2.5	Timer Complementary PWM functions.....	749
52.2.6	Timer Complementary One Pulse functions.....	750
52.2.7	Peripheral Control functions	750
52.2.8	Extended Callbacks functions	750
52.2.9	Extended Peripheral State functions	750

52.2.10	Detailed description of functions	751
52.3	TIMEx Firmware driver defines	762
52.3.1	TIMEx	762
53	HAL TSC Generic Driver	768
53.1	TSC Firmware driver registers structures.....	768
53.1.1	TSC_InitTypeDef	768
53.1.2	TSC_IOConfigTypeDef.....	769
53.1.3	TSC_HandleTypeDef	769
53.2	TSC Firmware driver API description	769
53.2.1	TSC specific features	769
53.2.2	How to use this driver	770
53.2.3	Initialization and de-initialization functions	770
53.2.4	IO Operation functions.....	770
53.2.5	Peripheral Control functions	771
53.2.6	State and Errors functions.....	771
53.2.7	Detailed description of functions	771
53.3	TSC Firmware driver defines.....	775
53.3.1	TSC.....	775
54	HAL UART Generic Driver.....	784
54.1	UART Firmware driver registers structures	784
54.1.1	UART_InitTypeDef	784
54.1.2	UART_AdvFeatureInitTypeDef.....	784
54.1.3	UART_WakeUpTypeDef	785
54.1.4	UART_HandleTypeDef.....	786
54.2	UART Firmware driver API description	787
54.2.1	How to use this driver	787
54.2.2	Initialization and Configuration functions.....	789
54.2.3	IO operation functions	790
54.2.4	Peripheral Control functions	790
54.2.5	Peripheral State and Error functions	790
54.2.6	Detailed description of functions	791
54.3	UART Firmware driver defines	802
54.3.1	UART	802
55	HAL UART Extension Driver.....	817
55.1	UARTEEx Firmware driver API description	817
55.1.1	UART peripheral extended features.....	817
55.1.2	Initialization and Configuration functions.....	817

55.1.3	IO operation function	817
55.1.4	Peripheral Control functions	817
55.1.5	Detailed description of functions	818
55.2	UARTEEx Firmware driver defines.....	820
55.2.1	UARTEx.....	820
56	HAL USART Generic Driver	821
56.1	USART Firmware driver registers structures.....	821
56.1.1	USART_InitTypeDef	821
56.1.2	USART_HandleTypeDef	821
56.2	USART Firmware driver API description	822
56.2.1	How to use this driver	822
56.2.2	Initialization and Configuration functions	824
56.2.3	IO operation functions	825
56.2.4	Peripheral State and Error functions	826
56.2.5	Detailed description of functions	827
56.3	USART Firmware driver defines.....	833
56.3.1	USART.....	833
57	HAL USART Extension Driver	842
57.1	USARTEEx Firmware driver defines	842
57.1.1	USARTEEx	842
58	HAL WWDG Generic Driver	843
58.1	WWDG Firmware driver registers structures.....	843
58.1.1	WWDG_InitTypeDef	843
58.1.2	WWDG_HandleTypeDef	843
58.2	WWDG Firmware driver API description	843
58.2.1	WWDG specific features	843
58.2.2	How to use this driver	844
58.2.3	Initialization and Configuration functions	845
58.2.4	IO operation functions	845
58.2.5	Detailed description of functions	845
58.3	WWDG Firmware driver defines.....	846
58.3.1	WWDG.....	846
59	LL ADC Generic Driver	850
59.1	ADC Firmware driver registers structures	850
59.1.1	LL_ADC_CommonInitTypeDef	850
59.1.2	LL_ADC_InitTypeDef.....	850

Contents	UM1786
59.1.3 LL_ADC_REG_InitTypeDef.....	851
59.1.4 LL_ADC_INJ_InitTypeDef	852
59.2 ADC Firmware driver API description.....	852
59.2.1 Detailed description of functions	852
59.3 ADC Firmware driver defines	950
59.3.1 ADC	950
60 LL BUS Generic Driver.....	991
60.1 BUS Firmware driver API description.....	991
60.1.1 Detailed description of functions	991
60.2 BUS Firmware driver defines	1005
60.2.1 BUS	1005
61 LL COMP Generic Driver.....	1007
61.1 COMP Firmware driver registers structures	1007
61.1.1 LL_COMP_InitTypeDef	1007
61.2 COMP Firmware driver API description	1008
61.2.1 Detailed description of functions	1008
61.3 COMP Firmware driver defines	1022
61.3.1 COMP	1022
62 LL CORTEX Generic Driver.....	1035
62.1 CORTEX Firmware driver API description	1035
62.1.1 Detailed description of functions	1035
62.2 CORTEX Firmware driver defines.....	1042
62.2.1 CORTEX.....	1042
63 LL CRC Generic Driver.....	1045
63.1 CRC Firmware driver API description	1045
63.1.1 Detailed description of functions	1045
63.2 CRC Firmware driver defines	1051
63.2.1 CRC	1051
64 LL DAC Generic Driver.....	1053
64.1 DAC Firmware driver registers structures	1053
64.1.1 LL_DAC_InitTypeDef.....	1053
64.2 DAC Firmware driver API description.....	1053
64.2.1 Detailed description of functions	1053
64.3 DAC Firmware driver defines	1071
64.3.1 DAC	1071

65	LL DMA Generic Driver	1077
65.1	DMA Firmware driver registers structures	1077
65.1.1	LL_DMA_InitTypeDef	1077
65.2	DMA Firmware driver API description	1078
65.2.1	Detailed description of functions	1078
65.3	DMA Firmware driver defines.....	1111
65.3.1	DMA.....	1111
66	LL EXTI Generic Driver	1116
66.1	EXTI Firmware driver registers structures	1116
66.1.1	LL_EXTI_InitTypeDef	1116
66.2	EXTI Firmware driver API description	1116
66.2.1	Detailed description of functions	1116
66.3	EXTI Firmware driver defines.....	1138
66.3.1	EXTI.....	1138
67	LL GPIO Generic Driver	1141
67.1	GPIO Firmware driver registers structures	1141
67.1.1	LL_GPIO_InitTypeDef	1141
67.2	GPIO Firmware driver API description	1141
67.2.1	Detailed description of functions	1141
67.3	GPIO Firmware driver defines.....	1157
67.3.1	GPIO.....	1157
68	LL HRTIM Generic Driver	1159
68.1	HRTIM Firmware driver API description.....	1159
68.1.1	Detailed description of functions	1159
68.2	HRTIM Firmware driver defines	1324
68.2.1	HRTIM	1324
69	LL I2C Generic Driver	1348
69.1	I2C Firmware driver registers structures	1348
69.1.1	LL_I2C_InitTypeDef.....	1348
69.2	I2C Firmware driver API description.....	1349
69.2.1	Detailed description of functions	1349
69.3	I2C Firmware driver defines	1390
69.3.1	I2C	1390
70	LL I2S Generic Driver	1395
70.1	I2S Firmware driver registers structures	1395

70.1.1	LL_I2S_InitTypeDef	1395
70.2	I2S Firmware driver API description	1395
70.2.1	Detailed description of functions	1395
70.3	I2S Firmware driver defines	1409
70.3.1	I2S	1409
71	LL IWDG Generic Driver	1412
71.1	IWDG Firmware driver API description	1412
71.1.1	Detailed description of functions	1412
71.2	IWDG Firmware driver defines	1416
71.2.1	IWDG	1416
72	LL OPAMP Generic Driver	1417
72.1	OPAMP Firmware driver registers structures	1417
72.1.1	LL_OPAMP_InitTypeDef	1417
72.2	OPAMP Firmware driver API description	1417
72.2.1	Detailed description of functions	1417
72.3	OPAMP Firmware driver defines	1429
72.3.1	OPAMP	1429
73	LL PWR Generic Driver	1434
73.1	PWR Firmware driver API description	1434
73.1.1	Detailed description of functions	1434
73.2	PWR Firmware driver defines	1440
73.2.1	PWR	1440
74	LL RCC Generic Driver	1443
74.1	RCC Firmware driver registers structures	1443
74.1.1	LL_RCC_ClocksTypeDef	1443
74.2	RCC Firmware driver API description	1443
74.2.1	Detailed description of functions	1443
74.3	RCC Firmware driver defines	1474
74.3.1	RCC	1474
75	LL RTC Generic Driver	1484
75.1	RTC Firmware driver registers structures	1484
75.1.1	LL_RTC_InitTypeDef	1484
75.1.2	LL_RTC_TimeTypeDef	1484
75.1.3	LL_RTC_DateTypeDef	1485
75.1.4	LL_RTC_AlarmTypeDef	1485
75.2	RTC Firmware driver API description	1486

75.2.1	Detailed description of functions	1486
75.3	RTC Firmware driver defines	1551
75.3.1	RTC	1551
76	LL SPI Generic Driver.....	1560
76.1	SPI Firmware driver registers structures	1560
76.1.1	LL_SPI_InitTypeDef	1560
76.2	SPI Firmware driver API description	1561
76.2.1	Detailed description of functions	1561
76.3	SPI Firmware driver defines	1583
76.3.1	SPI.....	1583
77	LL SYSTEM Generic Driver.....	1587
77.1	SYSTEM Firmware driver API description	1587
77.1.1	Detailed description of functions	1587
77.2	SYSTEM Firmware driver defines	1610
77.2.1	SYSTEM.....	1610
78	LL TIM Generic Driver	1614
78.1	TIM Firmware driver registers structures.....	1614
78.1.1	LL_TIM_InitTypeDef	1614
78.1.2	LL_TIM_OC_InitTypeDef.....	1614
78.1.3	LL_TIM_IC_InitTypeDef	1615
78.1.4	LL_TIM_ENCODER_InitTypeDef.....	1616
78.1.5	LL_TIM_HALLSENSOR_InitTypeDef.....	1617
78.1.6	LL_TIM_BDTR_InitTypeDef	1617
78.2	TIM Firmware driver API description	1619
78.2.1	Detailed description of functions	1619
78.3	TIM Firmware driver defines.....	1693
78.3.1	TIM.....	1693
79	LL USART Generic Driver	1708
79.1	USART Firmware driver registers structures.....	1708
79.1.1	LL_USART_InitTypeDef	1708
79.1.2	LL_USART_ClockInitTypeDef.....	1708
79.2	USART Firmware driver API description	1709
79.2.1	Detailed description of functions	1709
79.3	USART Firmware driver defines.....	1778
79.3.1	USART.....	1778

80 LL UTILS Generic Driver	1784
80.1 UTILS Firmware driver registers structures.....	1784
80.1.1 LL_UTILS_PLLInitTypeDef	1784
80.1.2 LL_UTILS_ClkInitTypeDef.....	1784
80.2 UTILS Firmware driver API description	1784
80.2.1 System Configuration functions.....	1784
80.2.2 Detailed description of functions	1785
80.3 UTILS Firmware driver defines.....	1788
80.3.1 UTILS.....	1788
81 LL WWDG Generic Driver	1789
81.1 WWDG Firmware driver API description	1789
81.1.1 Detailed description of functions	1789
81.2 WWDG Firmware driver defines.....	1793
81.2.1 WWDG.....	1793
82 Correspondence between API registers and API low-layer driver functions.....	1794
82.1 ADC	1794
82.2 BUS.....	1803
82.3 COMP	1811
82.4 CORTEX	1812
82.5 CRC	1813
82.6 DAC	1814
82.7 DMA.....	1816
82.8 EXTI.....	1819
82.9 GPIO	1820
82.10 I2C	1821
82.11 I2S.....	1825
82.12 IWDG	1826
82.13 OPAMP	1827
82.14 PWR.....	1828
82.15 RCC	1829
82.16 RTC.....	1833
82.17 SPI	1841
82.18 SYSTEM	1844
82.19 TIM.....	1852

82.20	USART.....	1862
82.21	WWDG.....	1870
83	FAQs.....	1871
84	Revision history	1875

List of tables

Table 1: Acronyms and definitions	28
Table 2: HAL driver files	30
Table 3: User-application files	31
Table 4: API classification	35
Table 5: List of devices supported by HAL drivers	36
Table 6: HAL API naming rules	39
Table 7: Macros handling interrupts and specific clock configurations	40
Table 8: Callback functions	41
Table 9: HAL generic APIs	42
Table 10: HAL extension APIs	43
Table 11: Define statements used for HAL configuration	47
Table 12: Description of GPIO_InitTypeDef structure	49
Table 13: Description of EXTI configuration macros	51
Table 14: MSP functions	56
Table 15: Timeout values	60
Table 16: LL driver files	64
Table 17: Common peripheral initialization functions	67
Table 18: Optional peripheral initialization functions	67
Table 19: Specific Interrupt, DMA request and status flags management	69
Table 20: Available function formats	69
Table 21: Peripheral clock activation/deactivation management	69
Table 22: Peripheral activation/deactivation management	70
Table 23: Peripheral configuration management	70
Table 24: Peripheral register management	70
Table 25: Correspondence between ADC registers and ADC low-layer driver functions	1794
Table 26: Correspondence between BUS registers and BUS low-layer driver functions	1803
Table 27: Correspondence between COMP registers and COMP low-layer driver functions	1811
Table 28: Correspondence between CORTEX registers and CORTEX low-layer driver functions	1812
Table 29: Correspondence between CRC registers and CRC low-layer driver functions	1813
Table 30: Correspondence between DAC registers and DAC low-layer driver functions	1814
Table 31: Correspondence between DMA registers and DMA low-layer driver functions	1816
Table 32: Correspondence between EXTI registers and EXTI low-layer driver functions	1819
Table 33: Correspondence between GPIO registers and GPIO low-layer driver functions	1820
Table 34: Correspondence between I2C registers and I2C low-layer driver functions	1821
Table 35: Correspondence between I2S registers and I2S low-layer driver functions	1825
Table 36: Correspondence between IWDG registers and IWDG low-layer driver functions	1826
Table 37: Correspondence between OPAMP registers and OPAMP low-layer driver functions	1827
Table 38: Correspondence between PWR registers and PWR low-layer driver functions	1828
Table 39: Correspondence between RCC registers and RCC low-layer driver functions	1829
Table 40: Correspondence between RTC registers and RTC low-layer driver functions	1833
Table 41: Correspondence between SPI registers and SPI low-layer driver functions	1841
Table 42: Correspondence between SYSTEM registers and SYSTEM low-layer driver functions	1844
Table 43: Correspondence between TIM registers and TIM low-layer driver functions	1852
Table 44: Correspondence between USART registers and USART low-layer driver functions	1862
Table 45: Correspondence between WWDG registers and WWDG low-layer driver functions	1870
Table 46: Document revision history	1875

List of figures

Figure 1: Example of project template	32
Figure 2: Adding device-specific functions	44
Figure 3: Adding family-specific functions	44
Figure 4: Adding new peripherals	45
Figure 5: Updating existing APIs	45
Figure 6: File inclusion model	46
Figure 7: HAL driver model	54
Figure 8: low-layer driver folders	65
Figure 9: Low-layer driver CMSIS files	66

1 Acronyms and definitions

Table 1: Acronyms and definitions

Acronym	Definition
ADC	Analog-to-digital converter
ANSI	American National Standards Institute
API	Application Programming Interface
BSP	Board Support Package
COMP	Comparator
CMSIS	Cortex Microcontroller Software Interface Standard
CPU	Central Processing Unit
CRYP	Cryptographic processor unit
CRC	CRC calculation unit
DAC	Digital to analog converter
DMA	Direct Memory Access
EXTI	External interrupt/event controller
FLASH	Flash memory
FMC	Flexible Memory Controller
GPIO	General purpose I/Os
HAL	Hardware abstraction layer
HRTIM	High Resolution Timer
I2C	Inter-integrated circuit
I2S	Inter-integrated sound
IRDA	InfraRed Data Association
IWDG	Independent watchdog
LCD	Liquid Crystal Display Controller
MSP	MCU Specific Package
NAND	NAND Flash memory
NOR	NOR Flash memory
NVIC	Nested Vectored Interrupt Controller
PCD	USB Peripheral Controller Driver
PWR	Power controller
RCC	Reset and clock controller
RNG	Random Number Generator
RTC	Real-time clock
SD	Secure Digital
SDADC	Sigma-delta Analog-to-digital Converter
SRAM	SRAM external memory

Acronym	Definition
SMARTCARD	Smartcard IC
SPI	Serial Peripheral interface
SysTick	System tick timer
TIM	Advanced-control, general-purpose or basic timer
TSC	Touch Sensing Controller
UART	Universal asynchronous receiver/transmitter
USART	Universal synchronous receiver/transmitter
WWDG	Window watchdog
USB	Universal Serial Bus
PPP	STM32 peripheral or block

2 Overview of HAL drivers

The HAL drivers were designed to offer a rich set of APIs and to interact easily with the application upper layers.

Each driver consists of a set of functions covering the most common peripheral features. The development of each driver is driven by a common API which standardizes the driver structure, the functions and the parameter names.

The HAL drivers include a set of driver modules, each module being linked to a standalone peripheral. However, in some cases, the module is linked to a peripheral functional mode. As an example, several modules exist for the USART peripheral: USART driver module, USART driver module, SMARTCARD driver module and IRDA driver module.

The HAL main features are the following:

- Cross-family portable set of APIs covering the common peripheral features as well as extension APIs in case of specific peripheral features.
- Three API programming models: polling, interrupt and DMA.
- APIs are RTOS compliant:
 - Fully reentrant APIs
 - Systematic usage of timeouts in polling mode.
- Support of peripheral multi-instance allowing concurrent API calls for multiple instances of a given peripheral (USART1, USART2...)
- All HAL APIs implement user-callback functions mechanism:
 - Peripheral Init/DeInit HAL APIs can call user-callback functions to perform peripheral system level Initialization/De-Initialization (clock, GPIOs, interrupt, DMA)
 - Peripherals interrupt events
 - Error events.
- Object locking mechanism: safe hardware access to prevent multiple spurious accesses to shared resources.
- Timeout used for all blocking processes: the timeout can be a simple counter or a timebase.

2.1 HAL and user-application files

2.1.1 HAL driver files

A HAL drivers are composed of the following set of files:

Table 2: HAL driver files

File	Description
<i>stm32f3xx_hal_ppp.c</i>	Main peripheral/module driver file. It includes the APIs that are common to all STM32 devices. <i>Example: stm32f3xx_hal_adc.c, stm32f3xx_hal_irda.c, ...</i>
<i>stm32xx_hal_ppp.h</i>	Header file of the main driver C file It includes common data, handle and enumeration structures, define statements and macros, as well as the exported generic APIs. <i>Example: stm32xx_hal_adc.h, stm32xx_hal_irda.h, ...</i>

File	Description
<i>stm32xx_hal_ppp_ex.c</i>	Extension file of a peripheral/module driver. It includes the specific APIs for a given part number or family, as well as the newly defined APIs that overwrite the default generic APIs if the internal process is implemented in different way. <i>Example: stm32xx_hal_adc_ex.c, stm32xx_hal_dma_ex.c, ...</i>
<i>stm32xx_hal_ppp_ex.h</i>	Header file of the extension C file. It includes the specific data and enumeration structures, define statements and macros, as well as the exported device part number specific APIs <i>Example: stm32xx_hal_adc_ex.h, stm32xx_hal_dma_ex.h, ...</i>
<i>stm32xx_hal.c</i>	This file is used for HAL initialization and contains DBGMCU, Remap and Time Delay based on systick APIs.
<i>stm32xx_hal.h</i>	xx_hal.c header file
<i>stm32xx_hal_msp_template.c</i>	Template file to be copied to the user application folder. It contains the MSP initialization and de-initialization (main routine and callbacks) of the peripheral used in the user application.
<i>stm32xx_hal_conf_template.h</i>	Template file allowing to customize the drivers for a given application.
<i>stm32xx_hal_def.h</i>	Common HAL resources such as common define statements, enumerations, structures and macros.

2.1.2 User-application files

The minimum files required to build an application using the HAL are listed in the table below:

Table 3: User-application files

File	Description
<i>system_stm32f3xx.c</i>	This file contains SystemInit() which is called at startup just after reset and before branching to the main program. It does not configure the system clock at startup (contrary to the standard library). This is to be done using the HAL APIs in the user files. It allows relocating the vector table in internal SRAM.
<i>startup_stm32f3xx.s</i>	Toolchain specific file that contains reset handler and exception vectors. For some toolchains, it allows adapting the stack/heap size to fit the application requirements.
<i>stm32f3xx_flash.icf (optional)</i>	Linker file for EWARM toolchain allowing mainly to adapt the stack/heap size to fit the application requirements.
<i>stm32f3xx_hal_msp.c</i>	This file contains the MSP initialization and de-initialization (main routine and callbacks) of the peripheral used in the user application.
<i>stm32f3xx_hal_conf.h</i>	This file allows the user to customize the HAL drivers for a specific application. It is not mandatory to modify this configuration. The application can use the default configuration without any modification.

File	Description
<i>stm32f3xx_it.c/h</i>	This file contains the exceptions handler and peripherals interrupt service routine, and calls HAL_IncTick() at regular time intervals to increment a local variable (declared in <i>stm32f3xx_hal.c</i>) used as HAL timebase. By default, this function is called each 1ms in SysTick ISR.. The PPP_IRQHandler() routine must call HAL_PPP_IRQHandler() if an interrupt based process is used within the application.
<i>main.c/h</i>	This file contains the main program routine, mainly: <ul style="list-style-type: none"> • the call to HAL_Init() • assert_failed() implementation • system clock configuration • peripheral HAL initialization and user application code.

The STM32Cube package comes with ready-to-use project templates, one for each supported board. Each project contains the files listed above and a preconfigured project for the supported toolchains.

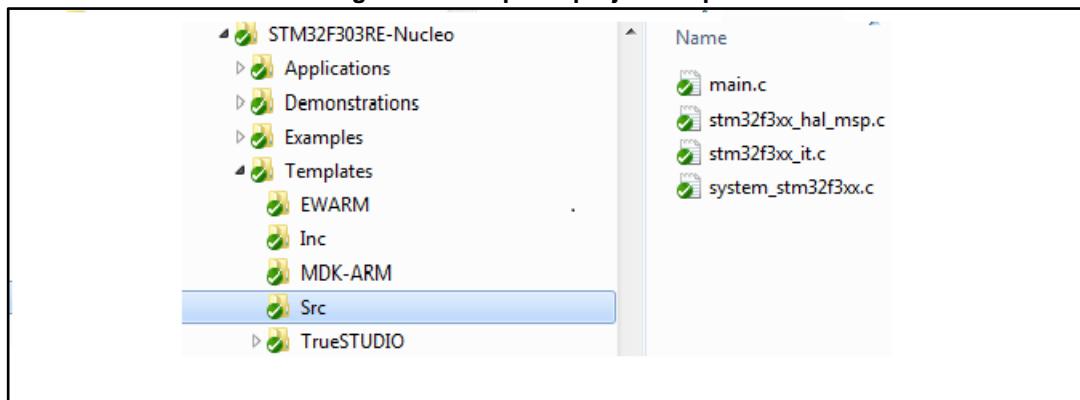
Each project template provides empty main loop function and can be used as a starting point to get familiar with project settings for STM32Cube. Its features are the following:

- It contains the sources of HAL, CMSIS and BSP drivers which are the minimal components to develop a code on a given board.
- It contains the include paths for all the firmware components.
- It defines the STM32 device supported, and allows configuring the CMSIS and HAL drivers accordingly.
- It provides ready-to-use user files preconfigured as defined below:
 - HAL is initialized
 - SysTick ISR implemented for HAL_Delay()
 - System clock configured with the maximum frequency of the device



If an existing project is copied to another location, then include paths must be updated.

Figure 1: Example of project template



2.2 HAL data structures

Each HAL driver can contain the following data structures:

- Peripheral handle structures
- Initialization and configuration structures
- Specific process structures.

2.2.1 Peripheral handle structures

The APIs have a modular generic multi-instance architecture that allows working with several IP instances simultaneously.

PPP_HandleTypeDef *handle is the main structure that is implemented in the HAL drivers. It handles the peripheral/module configuration and registers and embeds all the structures and variables needed to follow the peripheral device flow.

The peripheral handle is used for the following purposes:

- Multi instance support: each peripheral/module instance has its own handle. As a result instance resources are independent.
- Peripheral process intercommunication: the handle is used to manage shared data resources between the process routines.
Example: global pointers, DMA handles, state machine.
- Storage : this handle is used also to manage global variables within a given HAL driver.

An example of peripheral structure is shown below:

```
typedef struct
{
    USART_TypeDef *Instance; /* USART registers base address */
    USART_InitTypeDef Init; /* Usart communication parameters */
    uint8_t *pTxBuffPtr; /* Pointer to Usart Tx transfer Buffer */
    uint16_t TxXferSize; /* Usart Tx Transfer size */
    __IO uint16_t TxXferCount; /* Usart Tx Transfer Counter */
    uint8_t *pRxBuffPtr; /* Pointer to Usart Rx transfer Buffer */
    uint16_t RxXferSize; /* Usart Rx Transfer size */
    __IO uint16_t RxXferCount; /* Usart Rx Transfer Counter */
    DMA_HandleTypeDef *hdmatx; /* Usart Tx DMA Handle parameters */
    DMA_HandleTypeDef *hdmarx; /* Usart Rx DMA Handle parameters */
    HAL_LockTypeDef Lock; /* Locking object */
    __IO HAL_USART_StateTypeDef State; /* Usart communication state */
    __IO HAL_USART_ErrorTypeDef ErrorCode; /* USART Error code */
}USART_HandleTypeDef;
```



1) The multi-instance feature implies that all the APIs used in the application are re-entrant and avoid using global variables because subroutines can fail to be re-entrant if they rely on a global variable to remain unchanged but that variable is modified when the subroutine is recursively invoked. For this reason, the following rules are respected:

- Re-entrant code does not hold any static (or global) non-constant data: re-entrant functions can work with global data. For example, a re-entrant interrupt service routine can grab a piece of hardware status to work with (e.g. serial port read buffer) which is not only global, but volatile. Still, typical use of static variables and global data is not advised, in the sense that only atomic read-modify-write instructions should be used in these variables. It should not be possible for an interrupt or signal to occur during the execution of such an instruction.
- Reentrant code does not modify its own code.



2) When a peripheral can manage several processes simultaneously using the DMA (full duplex case), the DMA interface handle for each process is added in the PPP_HandleTypeDef.



3) For the shared and system peripherals, no handle or instance object is used. The peripherals concerned by this exception are the following:

- GPIO
- SYSTICK
- NVIC
- PWR
- RCC
- FLASH.

2.2.2 Initialization and configuration structure

These structures are defined in the generic driver header file when it is common to all part numbers. When they can change from one part number to another, the structures are defined in the extension header file for each part number.

```
typedef struct
{
    uint32_t BaudRate; /*!< This member configures the UART communication baudrate.*/
    uint32_t WordLength; /*!< Specifies the number of data bits transmitted or received
    in a frame.*/
    uint32_t StopBits; /*!< Specifies the number of stop bits transmitted.*/
    uint32_t Parity; /*!< Specifies the parity mode. */
    uint32_t Mode; /*!< Specifies whether the Receive or Transmit mode is enabled or
    disabled.*/
    uint32_t HwFlowCtl; /*!< Specifies whether the hardware flow control mode is enabled
    or disabled.*/
    uint32_t OverSampling; /*!< Specifies whether the Over sampling 8 is enabled or
    disabled,
    to achieve higher speed (up to fPCLK/8).*/
}UART_HandleTypeDef;
```



The config structure is used to initialize the sub-modules or sub-instances. See below example:

```
HAL_ADC_ConfigChannel (ADC_HandleTypeDef* hadc, ADC_ChannelConfTypeDef* sConfig)
```

2.2.3 Specific process structures

The specific process structures are used for specific process (common APIs). They are defined in the generic driver header file.

Example:

```
HAL_PPP_Process (PPP_HandleTypeDef* hadc, PPP_ProcessConfig* sConfig)
```

2.3 API classification

The HAL APIs are classified into three categories:

- **Generic APIs:** common generic APIs applying to all STM32 devices. These APIs are consequently present in the generic HAL driver files of all STM32 microcontrollers.

```

HAL_StatusTypeDef HAL_ADC_Init(ADC_HandleTypeDef* hadc); HAL_StatusTypeDef
HAL_ADC_DeInit(ADC_HandleTypeDef *hadc); HAL_StatusTypeDef
HAL_ADC_Start(ADC_HandleTypeDef* hadc); HAL_StatusTypeDef
HAL_ADC_Stop(ADC_HandleTypeDef* hadc); HAL_StatusTypeDef
HAL_ADC_Start_IT(ADC_HandleTypeDef* hadc); HAL_StatusTypeDef
HAL_ADC_Stop_IT(ADC_HandleTypeDef* hadc); void HAL_ADC_IRQHandler(ADC_HandleTypeDef* hadc);

```

- **Extension APIs:** This set of API is divided into two sub-categories :
 - **Family specific APIs:** APIs applying to a given family. They are located in the extension HAL driver file (see example below related to the ADC).

```

HAL_StatusTypeDef HAL_ADCEx_Calibration_Start(ADC_HandleTypeDef* hadc, uint32_t
SingleDiff); uint32_t HAL_ADCEx_Calibration_GetValue(ADC_HandleTypeDef* hadc,
uint32_t SingleDiff);

```

- **Device part number specific APIs:** These APIs are implemented in the extension file and delimited by specific define statements relative to a given part number.

```

#if defined(STM32F302xC) || defined(STM32F303xC) || defined(STM32F358xx) || \
defined(STM32F303x8) || defined(STM32F334x8) || defined(STM32F328xx) || \
defined(STM32F301x8) || defined(STM32F302x8) || defined(STM32F318xx) || \
defined(STM32F373xC) || defined(STM32F378xx) #endif /* STM32F302xC || STM32F303xC || \
STM32F358xx || */ /* STM32F303x8 || STM32F334x8 || STM32F328xx || */ /* STM32F301x8 \
|| STM32F302x8 || STM32F318xx */ /* STM32F373xC || STM32F378xx */

```



The data structure related to the specific APIs is delimited by the device part number define statement. It is located in the corresponding extension header C file.

The following table summarizes the location of the different categories of HAL APIs in the driver files.

Table 4: API classification

	Generic file	Extension file
Common APIs	X	X ⁽¹⁾
Family specific APIs		X
Device specific APIs		X

Notes:

⁽¹⁾In some cases, the implementation for a specific device part number may change . In this case the generic API is declared as weak function in the extension file. The API is implemented again to overwrite the default function



Family specific APIs are only related to a given family. This means that if a specific API is implemented in another family, and the arguments of this latter family are different, additional structures and arguments might need to be added.



The IRQ handlers are used for common and family specific processes.

2.4 Devices supported by HAL drivers

Table 5: List of devices supported by HAL drivers

IP/Module	STM32F301x6/x8	STM32F302x6/x8	STM32F302xB/xC	STM32F302xE	STM32F303x6/x8	STM32F303xC	STM32F303xE	STM32F303xBxC	STM32F303x6/x8	STM32F318xx	STM32F328xx	STM32F358xx	STM32F378xx	STM32F398xx
stm32f3xx_hal.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_adc.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_adc_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_can.c	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
stm32f3xx_hal_cec.c	No	No	No	No	No	No	No	Yes	No	No	No	No	Yes	No
stm32f3xx_hal_comp.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_cortex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_crc.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_crc_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_dac.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_dac_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_dma.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_flash.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_flash_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_gpio.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_hrtim.c	No	No	No	No	No	No	No	No	Yes	No	Yes	No	No	No
stm32f3xx_hal_i2c.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_i2c_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_i2s.c	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes
stm32f3xx_hal_i2s_ex.c	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes
stm32f3xx_hal_irda.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_iwdg.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

IP/Module	STM32 F301x6 /x8	STM32F3 02x6/x8	STM32F30 2xB/xC	STM32F 302xE	STM32F3 03x6/x8	STM32F 303xC	STM32F 303xE	STM32F37 3xB/xC	STM32F3 34x6/x8	STM32F 318xx	STM32F 328xx	STM32F 358xx	STM32F 378xx	STM32F 398xx
stm32f3xx_hal_msp_temp_late.c	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
stm32f3xx_hal_nand.c	No	No	No	Yes	No	No	Yes	No	No	No	No	No	No	Yes
stm32f3xx_hal_nor.c	No	No	No	Yes	No	No	Yes	No	No	No	No	No	No	Yes
stm32f3xx_hal_opamp.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes
stm32f3xx_hal_opamp_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes
stm32f3xx_hal_pcd.c	No	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	No	No	No	Yes
stm32f3xx_hal_pcd_ex.c	No	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	No	No	No	Yes
stm32f3xx_hal_pccard.c	No	No	No	Yes	No	No	Yes	No	No	No	No	No	No	Yes
stm32f3xx_hal_pwr.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_pwr_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	Yes	Yes
stm32f3xx_hal_rcc.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_rcc_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_rtc.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_rtc_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_sdadc.c	No	No	No	No	No	No	No	Yes	No	No	No	No	Yes	No
stm32f3xx_hal_smartcard.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_smartcard_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_smbus.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_spi.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_sram.c	No	No	No	Yes	No	No	Yes	No	No	No	No	No	No	Yes

Overview of HAL drivers

UM1786

IP/Module	STM32 F301x6 /x8	STM32F3 02x6/x8	STM32F30 2xB/xC	STM32F 302xE	STM32F3 03x6/x8	STM32F 303xC	STM32F 303xE	STM32F37 3xB/xC	STM32F3 34x6/x8	STM32F 318xx	STM32F 328xx	STM32F 358xx	STM32F 378xx	STM32F 398xx
stm32f3xx_hal_tim.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_tim_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_tsc.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_uart.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_uart_ex.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_usart.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_hal_wwdg.c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
stm32f3xx_ll_fmc.c	No	No	No	Yes	No	No	Yes	No	No	No	No	No	No	Yes

2.5 HAL driver rules

2.5.1 HAL API naming rules

The following naming rules are used in HAL drivers:

Table 6: HAL API naming rules

	Generic	Family specific	Device specific
File names	<i>stm32f3xx_hal_ppp (c/h)</i>	<i>stm32f3xx_hal_ppp_ex (c/h)</i>	<i>stm32f3xx_hal_ppp_ex (c/h)</i>
Module name	<i>HAL_PPP_MODULE</i>		
Function name	<i>HAL_PPP_Function</i> <i>HAL_PPP_FeatureFunction_MODE</i>	<i>HAL_PPPEx_Function</i> <i>HAL_PPPEx_FeatureFunction_MODE</i>	<i>HAL_PPPEx_Function</i> <i>HAL_PPPEx_FeatureFunction_MODE</i>
Handle name	<i>PPP_HandleTypeDef</i>	NA	NA
Init structure name	<i>PPP_InitTypeDef</i>	NA	<i>PPP_InitTypeDef</i>
Enum name	<i>HAL_PPP_StructnameTypeDef</i>	NA	NA

- The **PPP** prefix refers to the peripheral functional mode and not to the peripheral itself. For example, if the USART, PPP can be USART, IRDA, UART or SMARTCARD depending on the peripheral mode.
- The constants used in one file are defined within this file. A constant used in several files is defined in a header file. All constants are written in uppercase, except for peripheral driver function parameters.
- typedef variable names should be suffixed with _TypeDef.
- Registers are considered as constants. In most cases, their name is in uppercase and uses the same acronyms as in the STM32F3xx reference manuals.
- Peripheral registers are declared in the PPP_TypeDef structure (e.g. ADC_TypeDef) in the CMSIS header file corresponding to the selected platform: stm32f301x8.h, stm32f302x8.h, stm32f302xc.h, stm32f302xe.h, stm32f303x8.h, stm32f303xc.h, stm32f303xe.h, stm32f318xx.h, stm32f328xx.h, stm32f334x8.h, stm32f358xx.h, stm32f373xc.h, stm32f378xx.h and stm32f398xx.h . The platform is selected by enabling the compilation switch in the compilation toolchain directive or in the stm32f3xx.h file.
- Peripheral function names are prefixed by HAL_, then the corresponding peripheral acronym in uppercase followed by an underscore. The first letter of each word is in uppercase (e.g. HAL_UART_Transmit()). Only one underscore is allowed in a function name to separate the peripheral acronym from the rest of the function name.
- The structure containing the PPP peripheral initialization parameters are named PPP_InitTypeDef (e.g. ADC_InitTypeDef).
- The structure containing the Specific configuration parameters for the PPP peripheral are named PPP_xxxxConfTypeDef (e.g. ADC_ChannelConfTypeDef).
- Peripheral handle structures are named PPP_HandleTypeDef (e.g DMA_HandleTypeDef)
- The functions used to initialize the PPP peripheral according to parameters specified in PPP_InitTypeDef are named HAL_PPP_Init (e.g. HAL_TIM_Init()).

- The functions used to reset the PPP peripheral registers to their default values are named PPP_Delnit, e.g. TIM_Delnit.
- The **MODE** suffix refers to the process mode, which can be polling, interrupt or DMA. As an example, when the DMA is used in addition to the native resources, the function should be called: *HAL_PPP_Function_DMA ()*.
- The **Feature** prefix should refer to the new feature. Example: *HAL_ADC_Start()* refers to the injection mode

2.5.2 HAL general naming rules

- For the shared and system peripherals, no handle or instance object is used. This rule applies to the following peripherals:
 - GPIO
 - SYSTICK
 - NVIC
 - RCC
 - FLASH.

Example: The *HAL_GPIO_Init()* requires only the GPIO address and its configuration parameters.

```
HAL_StatusTypeDef HAL_GPIO_Init (GPIO_TypeDef* GPIOx, GPIO_InitTypeDef *Init)
{
/*GPIO Initialization body */
}
```

- The macros that handle interrupts and specific clock configurations are defined in each peripheral/module driver. These macros are exported in the peripheral driver header files so that they can be used by the extension file. The list of these macros is defined below: This list is not exhaustive and other macros related to peripheral features can be added, so that they can be used in the user application.

Table 7: Macros handling interrupts and specific clock configurations

Macros	Description
<code>_HAL_PPP_ENABLE_IT(_HANDLE_, _INTERRUPT_)</code>	Enables a specific peripheral interrupt
<code>_HAL_PPP_DISABLE_IT(_HANDLE_, _INTERRUPT_)</code>	Disables a specific peripheral interrupt
<code>_HAL_PPP_GET_IT (_HANDLE_, _INTERRUPT_)</code>	Gets a specific peripheral interrupt status
<code>_HAL_PPP_CLEAR_IT (_HANDLE_, _INTERRUPT_)</code>	Clears a specific peripheral interrupt status
<code>_HAL_PPP_GET_FLAG (_HANDLE_, _FLAG_)</code>	Gets a specific peripheral flag status
<code>_HAL_PPP_CLEAR_FLAG (_HANDLE_, _FLAG_)</code>	Clears a specific peripheral flag status
<code>_HAL_PPP_ENABLE(_HANDLE_)</code>	Enables a peripheral
<code>_HAL_PPP_DISABLE(_HANDLE_)</code>	Disables a peripheral
<code>_HAL_PPP_XXXX (_HANDLE_, _PARAM_)</code>	Specific PPP HAL driver macro
<code>_HAL_PPP_GET_IT_SOURCE (_HANDLE_, _INTERRUPT_)</code>	Checks the source of specified interrupt

- NVIC and SYSTICK are two ARM Cortex core features. The APIs related to these features are located in the `stm32f3xx_hal_cortex.c` file.
- When a status bit or a flag is read from registers, it is composed of shifted values depending on the number of read values and of their size. In this case, the returned status width is 32 bits. Example : `STATUS = XX | (YY << 16) or STATUS = XX | (YY << 8) | (YY << 16) | (YY << 24)`".
- The PPP handles are valid before using the `HAL_PPP_Init()` API. The init function performs a check before modifying the handle fields.

```
HAL_PPP_Init(PPP_HandleTypeDef)
{
    if(hppp == NULL)
    {
        return HAL_ERROR;
    }
}
```

- The macros defined below are used:
 - Conditional macro:

```
#define ABS(x) ((x) > 0) ? (x) : -(x)
```

- Pseudo-code macro (multiple instructions macro):

```
#define HAL_LINKDMA( HANDLE , PPP_DMA_FIELD , DMA_HANDLE ) \
do{ \
    ( __HANDLE__ )-> __PPP_DMA_FIELD__ = &( __DMA_HANDLE__ ); \
    ( __DMA_HANDLE__ ).Parent = ( __HANDLE__ ); \
} while(0)
```

2.5.3 HAL interrupt handler and callback functions

Besides the APIs, HAL peripheral drivers include:

- `HAL_PPP_IRQHandler()` peripheral interrupt handler that should be called from `stm32f3xx_it.c`
- User callback functions.

The user callback functions are defined as empty functions with “weak” attribute. They have to be defined in the user code.

There are three types of user callbacks functions:

- Peripheral system level initialization/ de-Initialization callbacks: `HAL_PPP_MspInit()` and `HAL_PPP_MspDeInit()`
- Process complete callbacks : `HAL_PPP_ProcessCpltCallback`
- Error callback: `HAL_PPP_ErrorCallback`.

Table 8: Callback functions

Callback functions	Example
<code>HAL_PPP_MspInit() / _DeInit()</code>	Ex: <code>HAL_USART_MspInit()</code> Called from <code>HAL_PPP_Init()</code> API function to perform peripheral system level initialization (GPIOs, clock, DMA, interrupt)
<code>HAL_PPP_ProcessCpltCallback</code>	Ex: <code>HAL_USART_TxCpltCallback</code> Called by peripheral or DMA interrupt handler when the process completes
<code>HAL_PPP_ErrorCallback</code>	Ex: <code>HAL_USART_ErrorCallback</code> Called by peripheral or DMA interrupt handler when an error occurs

2.6 HAL generic APIs

The generic APIs provide common generic functions applying to all STM32 devices. They are composed of four APIs groups:

- **Initialization and de-initialization functions:** HAL_PPP_Init(), HAL_PPP_DeInit()
- **IO operation functions:** HAL_PPP_Read(), HAL_PPP_Write(), HAL_PPP_Transmit(), HAL_PPP_Receive()
- **Control functions:** HAL_PPP_Set(), HAL_PPP_Get().
- **State and Errors functions:** HAL_PPP_GetState(), HAL_PPP_GetError().

For some peripheral/module drivers, these groups are modified depending on the peripheral/module implementation.

Example: in the timer driver, the API grouping is based on timer features (PWM, OC, IC...).

The initialization and de-initialization functions allow initializing a peripheral and configuring the low-level resources, mainly clocks, GPIO, alternate functions (AF) and possibly DMA and interrupts. The *HAL_DeInit()* function restores the peripheral default state, frees the low-level resources and removes any direct dependency with the hardware.

The IO operation functions perform a row access to the peripheral payload data in write and read modes.

The control functions are used to change dynamically the peripheral configuration and set another operating mode.

The peripheral state and errors functions allow retrieving in runtime the peripheral and data flow states, and identifying the type of errors that occurred. The example below is based on the ADC peripheral. The list of generic APIs is not exhaustive. It is only given as an example.

Table 9: HAL generic APIs

Function group	Common API name	Description
<i>Initialization group</i>	<i>HAL_ADC_Init()</i>	This function initializes the peripheral and configures the low -level resources (clocks, GPIO, AF..)
	<i>HAL_ADC_DeInit()</i>	This function restores the peripheral default state, frees the low-level resources and removes any direct dependency with the hardware.
<i>IO operation group</i>	<i>HAL_ADC_Start()</i>	This function starts ADC conversions when the polling method is used
	<i>HAL_ADC_Stop()</i>	This function stops ADC conversions when the polling method is used
	<i>HAL_ADC_PollForConversion()</i>	This function allows waiting for the end of conversions when the polling method is used. In this case, a timeout value is specified by the user according to the application.
	<i>HAL_ADC_Start_IT()</i>	This function starts ADC conversions when the interrupt method is used
	<i>HAL_ADC_Stop_IT()</i>	This function stops ADC conversions when the interrupt method is used
	<i>HAL_ADC_IRQHandler()</i>	This function handles ADC interrupt requests

Function group	Common API name	Description
<i>Control group</i>	<i>HAL_ADC_ConvCpltCallback()</i>	Callback function called in the IT subroutine to indicate the end of the current process or when a DMA transfer has completed
	<i>HAL_ADC_ErrorCallback()</i>	Callback function called in the IT subroutine if a peripheral error or a DMA transfer error occurred
<i>Control group</i>	<i>HAL_ADC_ConfigChannel()</i>	This function configures the selected ADC regular channel, the corresponding rank in the sequencer and the sample time
	<i>HAL_ADC_AnalogWDGConfig</i>	This function configures the analog watchdog for the selected ADC
<i>State and Errors group</i>	<i>HAL_ADC_GetState()</i>	This function allows getting in runtime the peripheral and the data flow states.
	<i>HAL_ADC_GetError()</i>	This function allows getting in runtime the error that occurred during IT routine

2.7 HAL extension APIs

2.7.1 HAL extension model overview

The extension APIs provide specific functions or overwrite modified APIs for a specific family (series) or specific part number within the same family.

The extension model consists of an additional file, `stm32f3xx_hal_ppp_ex.c`, that includes all the specific functions and define statements (`stm32f3xx_hal_ppp_ex.h`) for a given part number.

Below an example based on the ADC peripheral:

Table 10: HAL extension APIs

Function Group	Common API Name
<i>HAL_ADCEx_Calibration_Start()</i>	This function is used to start the automatic ADC calibration
<i>HAL_ADCEx_Calibration_GetValue()</i>	This function is used to get the ADC calibration factor
<i>HAL_ADCEx_Calibration_SetValue()</i>	This function is used to set the calibration factor to overwrite automatic conversion result

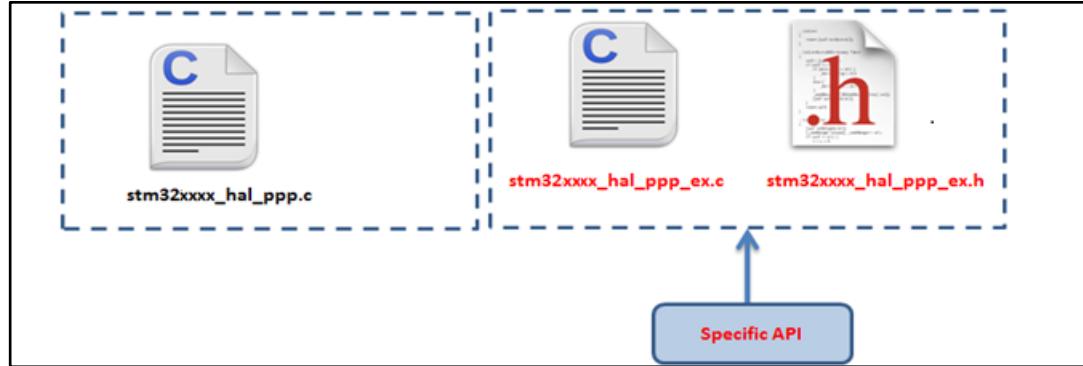
2.7.2 HAL extension model cases

The specific IP features can be handled by the HAL drivers in five different ways. They are described below.

Case 1: Adding a part number-specific function

When a new feature specific to a given device is required, the new APIs are added in the `stm32f3xx_hal_adc_ex.c` extension file. They are named `HAL_PPPEx_Function()`.

Figure 2: Adding device-specific functions



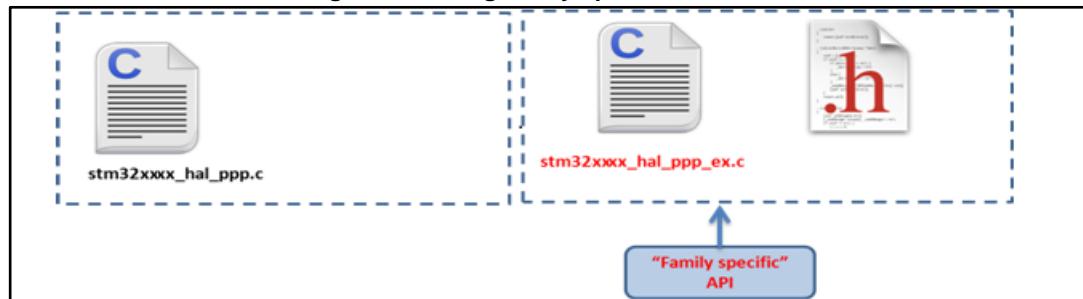
Example: `stm32f3xx_hal_adc_ex.c/h`

```
#if defined(STM32F302xE) || defined(STM32F303xE) || defined(STM32F398xx) || \
defined(STM32F302xC) || defined(STM32F303xC) || defined(STM32F358xx) || \
defined(STM32F303x8) || defined(STM32F334x8) || defined(STM32F328xx) || \
defined(STM32F301x8) || defined(STM32F302x8) || defined(STM32F318xx)
HAL_StatusTypeDef HAL_ADCEx_Calibration_Start(struct ADC_HandleTypeDef* hadc,
uint32_t SingleDiff);
uint32_t HAL_ADCEx_Calibration_GetValue(struct ADC_HandleTypeDef *hadc, uint32_t
SingleDiff);
HAL_StatusTypeDef HAL_ADCEx_Calibration_SetValue(struct ADC_HandleTypeDef *hadc,
uint32_t SingleDiff, uint32_t CalibrationFactor);
#endif /* STM32F302xE || STM32F303xE || STM32F398xx || */
/* STM32F302xC || STM32F303xC || STM32F358xx || */
/* STM32F303x8 || STM32F334x8 || STM32F328xx || */
/* STM32F301x8 || STM32F302x8 || STM32F318xx */
```

Case 2: Adding a family-specific function

In this case, the API is added in the extension driver C file and named `HAL_PPPEx_Function()`.

Figure 3: Adding family-specific functions

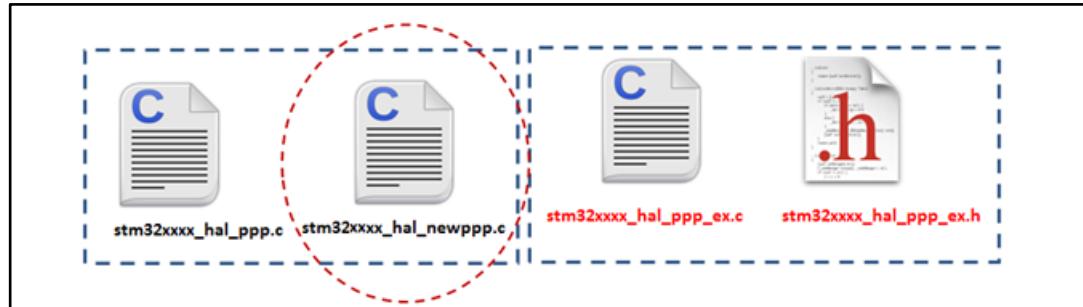


Case 3: Adding a new peripheral (specific to a device belonging to a given family)

When a peripheral which is available only in a specific device is required, the APIs corresponding to this new peripheral/module are added in `stm32f3xx_hal_newppp.c`. However the inclusion of this file is selected in the `stm32f3xx_hal_conf.h` using the macro:

```
#define HAL_NEWPPP_MODULE_ENABLED
```

Figure 4: Adding new peripherals

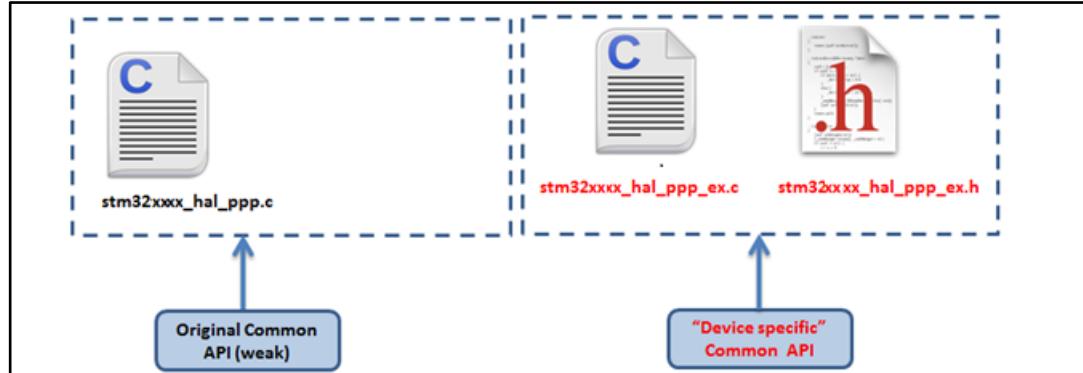


Example: xx_hal_sdadc.c/h

Case 4: Updating existing common APIs

In this case, the routines are defined with the same names in the `xx_hal_ppp_ex.c` extension file, while the generic API is defined as *weak*, so that the compiler will overwrite the original routine by the new defined function.

Figure 5: Updating existing APIs



Case 5: Updating existing data structures

The data structure for a specific device part number (e.g. `PPP_InitTypeDef`) can be composed of different fields. In this case, the data structure is defined in the extension header file and delimited by the specific part number define statement.

Example:

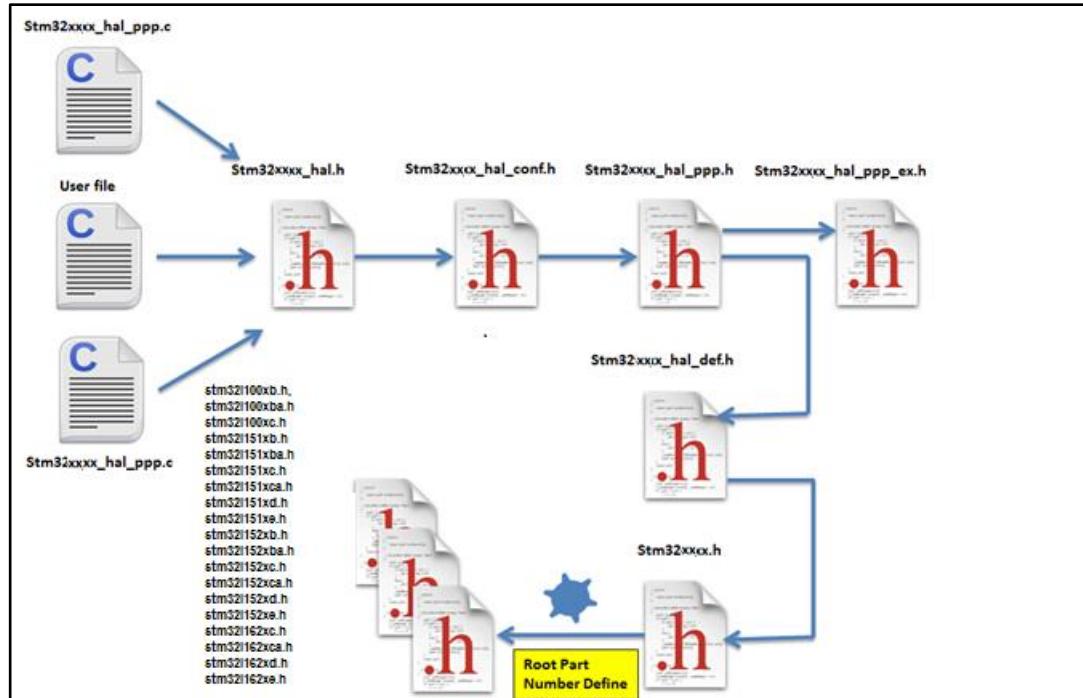
```
#if defined(STM32F373xC) || defined(STM32F378xx)
typedef struct
{
(...)

}PPP_InitTypeDef;
#endif /* STM32F373xC || STM32F378xx */
```

2.8 File inclusion model

The header of the common HAL driver file (stm32f3xx_hal.h) includes the common configurations for the whole HAL library. It is the only header file that is included in the user sources and the HAL C sources files to be able to use the HAL resources.

Figure 6: File inclusion model



A PPP driver is a standalone module which is used in a project. The user must enable the corresponding USE_HAL_PPP_MODULE define statement in the configuration file.

```
/*
 * @file stm32f3xx_hal_conf.h
 * @author MCD Application Team
 * @version VX.Y.Z * @date dd-mm-yyyy
 * @brief This file contains the modules to be used
 */
(...)

#define USE_HAL_USART_MODULE
#define USE_HAL_IRDA_MODULE
#define USE_HAL_DMA_MODULE
#define USE_HAL_RCC_MODULE
(...)
```

2.9 HAL common resources

The common HAL resources, such as common define enumerations, structures and macros, are defined in *stm32f3xx_hal_def.h*. The main common define enumeration is *HAL_StatusTypeDef*.

- **HAL Status** The HAL status is used by almost all HAL APIs, except for boolean functions and IRQ handler. It returns the status of the current API operations. It has four possible values as described below: **Service statut**

```
Typedef enum
{
    HAL_OK = 0x00, HAL_ERROR = 0x01, HAL_BUSY = 0x02, HAL_TIMEOUT = 0x03
} HAL_StatusTypeDef;
```

- **HAL Locked**

The HAL lock is used by all HAL APIs to prevent accessing by accident shared resources.

```
typedef enum
{
    HAL_UNLOCKED = 0x00, /*!<Resources unlocked */
    HAL_LOCKED = 0x01 /*!< Resources locked */
} HAL_LockTypeDef;
```

In addition to common resources, the `stm32f3xx_hal_def.h` file calls the `stm32f3xx.h` file in CMSIS library to get the data structures and the address mapping for all peripherals:

- Declarations of peripheral registers and bits definition.
- Macros to access peripheral registers hardware (Write register, Read register...etc.).

- **Common macros**

- Macro defining `HAL_MAX_DELAY`

```
#define HAL_MAX_DELAY 0xFFFFFFFF
– Macro linking a PPP peripheral to a DMA structure pointer: __HAL_LINKDMA();
```

```
#define HAL_LINKDMA( HANDLE , PPP_DMA_FIELD , DMA_HANDLE ) \
do{ \
    ( __HANDLE__ )->__PPP_DMA_FIELD__ = &( __DMA_HANDLE__ ); \
    ( __DMA_HANDLE__ ).Parent = ( __HANDLE__ ); \
} while(0)
```

2.10 HAL configuration

The configuration file, `stm32f3xx_hal_conf.h`, allows customizing the drivers for the user application. Modifying this configuration is not mandatory: the application can use the default configuration without any modification.

To configure these parameters, the user should enable, disable or modify some options by uncommenting, commenting or modifying the values of the related define statements as described in the table below:

Table 11: Define statements used for HAL configuration

Configuration item	Description	Default Value
<code>HSE_VALUE</code>	Defines the value of the external oscillator (HSE) expressed in Hz. The user must adjust this define statement when using a different crystal value.	8 000 000 (Hz)
<code>HSE_STARTUP_TIMEOUT</code>	Timeout for HSE start-up, expressed in ms	5000
<code>HSI_VALUE</code>	Defines the value of the internal oscillator (HSI) expressed in Hz.	16 000 000 (Hz)
<code>HSI_STARTUP_TIMEOUT</code>	Timeout for HSI start-up, expressed in ms	5000
<code>LSE_VALUE</code>	Defines the value of the external oscillator (HSE) expressed in Hz. The user must adjust this define statement when using a different crystal value.	32768 (Hz)
<code>LSE_STARTUP_TIMEOUT</code>	Timeout for LSE start-up, expressed in ms	5000
<code>LSI_VALUE</code>	Defines the value of the Internal Low Speed oscillator expressed in Hz. The real value may vary depending on the variations in voltage and temperature.	40 000 (Hz)
<code>VDD_VALUE</code>	VDD value	3300 (mV)

Configuration item	Description	Default Value
USERTOS	Enables the use of RTOS	FALSE (for future use)
PREFETCH_ENABLE	Enables prefetch feature	TRUE



The `stm32f3xx_hal_conf_template.h` file is located in the HAL drivers `/Inc` folder. It should be copied to the user folder, renamed and modified as described above.



By default, the values defined in the `stm32f3xx_hal_conf_template.h` file are the same as the ones used for the examples and demonstrations. All HAL include files are enabled so that they can be used in the user code without modifications.

2.11 HAL system peripheral handling

This chapter gives an overview of how the system peripherals are handled by the HAL drivers. The full API list is provided within each peripheral driver description section.

2.11.1 Clock

Two main functions can be used to configure the system clock:

- `HAL_RCC_OscConfig (RCC_OscInitTypeDef *RCC_OscInitStruct)`. This function configures/enables multiple clock sources (HSE, HSI, LSE, LSI, PLL).
- `HAL_RCC_ClockConfig (RCC_ClkInitTypeDef *RCC_ClkInitStruct, uint32_t FLatency)`. This function
 - selects the system clock source
 - configures AHB and APB clock dividers
 - configures the number of Flash memory wait states
 - updates the SysTick configuration when HCLK clock changes.

Some peripheral clocks are not derived from the system clock (RTC, USB...). In this case, the clock configuration is performed by an extended API defined in `stm32f3xx_hal_rcc_ex.c`: `HAL_RCCE_PeriphCLKConfig(RCC_PeriphCLKInitTypeDef *PeriphClkInit)`.

Additional RCC HAL driver functions are available:

- `HAL_RCC_DeInit()` Clock de-init function that return clock configuration to reset state
- Get clock functions that allow retrieving various clock configurations (system clock, HCLK, PCLK1, ...)
- MCO and CSS configuration functions

A set of macros are defined in `stm32f3xx_hal_rcc.h` and `stm32f3xx_hal_rcc_ex.h`. They allow executing elementary operations on RCC block registers, such as peripherals clock gating/reset control:

- `__PPP_CLK_ENABLE/__PPP_CLK_DISABLE` to enable/disable the peripheral clock
- `__PPP_FORCE_RESET/__PPP_RELEASE_RESET` to force/release peripheral reset
- `__PPP_CLK_SLEEP_ENABLE/__PPP_CLK_SLEEP_DISABLE` to enable/disable the peripheral clock during low power (Sleep) mode.

2.11.2 GPIOs

GPIO HAL APIs are the following:

- HAL_GPIO_Init() / HAL_GPIO_DeInit()
- HAL_GPIO_ReadPin() / HAL_GPIO_WritePin()
- HAL_GPIO_TogglePin () .

In addition to standard GPIO modes (input, output, analog), the pin mode can be configured as EXTI with interrupt or event generation.

When selecting EXTI mode with interrupt generation, the user must call HAL_GPIO_EXTI_IRQHandler() from stm32f3xx_it.c and implement HAL_GPIO_EXTI_Callback()

The table below describes the GPIO_InitTypeDef structure field.

Table 12: Description of GPIO_InitTypeDef structure

Structure field	Description
Pin	Specifies the GPIO pins to be configured. Possible values: GPIO_PIN_x or GPIO_PIN_All, where x[0..15]
Mode	Specifies the operating mode for the selected pins: GPIO mode or EXTI mode. Possible values are: <ul style="list-style-type: none"> • <u>GPIO mode</u> <ul style="list-style-type: none"> – GPIO_MODE_INPUT : Input floating – GPIO_MODE_OUTPUT_PP : Output push-pull – GPIO_MODE_OUTPUT_OD : Output open drain – GPIO_MODE_AF_PP : Alternate function push-pull – GPIO_MODE_AF_OD : Alternate function open drain – GPIO_MODE_ANALOG : Analog mode • <u>External Interrupt mode</u> <ul style="list-style-type: none"> – GPIO_MODE_IT_RISING : Rising edge trigger detection – GPIO_MODE_IT_FALLING : Falling edge trigger detection – GPIO_MODE_IT_RISING_FALLING : Rising/Falling edge trigger detection • <u>External Event mode</u> <ul style="list-style-type: none"> – GPIO_MODE_EVT_RISING : Rising edge trigger detection – GPIO_MODE_EVT_FALLING : Falling edge trigger detection – GPIO_MODE_EVT_RISING_FALLING: Rising/Falling edge trigger detection
Pull	Specifies the Pull-up or Pull-down activation for the selected pins. Possible values are: GPIO_NOPULL GPIO_PULLUP GPIO_PULLDOWN
Speed	Specifies the speed for the selected pins Possible values are: GPIO_SPEED_FREQ_LOW GPIO_SPEED_FREQ_MEDIUM GPIO_SPEED_FREQ_HIGH

Structure field	Description
Alternate	<p>Peripheral to be connected to the selected pins. Possible values: GPIO_AFx_PP, where AFx: is the alternate function index PP: is the peripheral instance Example: use GPIO_AF1_TIM2 to connect TIM2 IOs on AF1. These values are defined in the GPIO extended driver, since the AF mapping may change between product lines.</p>  <p>Refer to the "Alternate function mapping" table in the datasheets for the detailed description of the system and peripheral I/O alternate functions.</p>

Please find below typical GPIO configuration examples:

- Configuring GPIOs as output push-pull to drive external LEDs

```
GPIO_InitStruct.Pin = GPIO_PIN_12 | GPIO_PIN_13 | GPIO_PIN_14 | GPIO_PIN_15;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_PULLUP;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_MEDIUM;
HAL_GPIO_Init(GPIOD, &GPIO_InitStruct);
```

- Configuring PA0 as external interrupt with falling edge sensitivity:

```
GPIO_InitStructure.Mode = GPIO_MODE_IT_FALLING;
GPIO_InitStructure.Pull = GPIO_NOPULL;
GPIO_InitStructure.Pin = GPIO_PIN_0;
HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);
```

- Configuring USART1 Tx (PA9, mapped on AF4) as alternate function:

```
GPIO_InitStruct.Pin = GPIO_PIN_9;
GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
GPIO_InitStruct.Pull = GPIO_PULLUP;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
GPIO_InitStruct.Alternate = GPIO_AF4_USART1;
HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
```

2.11.3 Cortex NVIC and SysTick timer

The Cortex HAL driver, `stm32f3xx_hal_cortex.c`, provides APIs to handle NVIC and Systick. The supported APIs include:

- HAL_NVIC_SetPriority()
- HAL_NVIC_EnableIRQ() / HAL_NVIC_DisableIRQ()
- HAL_NVIC_SystemReset()
- HAL_SYSTICK_IRQHandler()
- HAL_NVIC_GetPendingIRQ() / HAL_NVIC_SetPendingIRQ() / HAL_NVIC_ClearPendingIRQ()
- HAL_SYSTICK_Config()
- HAL_SYSTICK_CLKSourceConfig()
- HAL_SYSTICK_Callback()

2.11.4 PWR

The PWR HAL driver handles power management. The features shared between all STM32 Series are listed below:

- PVD configuration, enabling/disabling and interrupt handling
 - HAL_PWR_PVDCConfig()
 - HAL_PWR_EnablePVD() / HAL_PWR_DisablePVD()
 - HAL_PWR_PVD_IRQHandler()
 - HAL_PWR_PVDCallback()
- Wakeup pin configuration
 - HAL_PWR_EnableWakeUpPin() / HAL_PWR_DisableWakeUpPin()
- Low-power mode entry
 - HAL_PWR_EnterSLEEPMode()
 - HAL_PWR_EnterSTOPMode()
 - HAL_PWR_EnterSTANDBYMode()
- Backup domain configuration
 - HAL_PWR_EnableBkUpAccess() / HAL_PWR_DisableBkUpAccess()

2.11.5 EXTI

The EXTI is not considered as a standalone peripheral but rather as a service used by other peripheral. As a result there are no EXTI APIs but each peripheral HAL driver implements the associated EXTI configuration and EXTI function are implemented as macros in its header file.

The first 16 EXTI lines connected to the GPIOs are managed within the GPIO driver. The GPIO_InitTypeDef structure allows configuring an I/O as external interrupt or external event.

The EXTI lines connected internally to the PVD, RTC, USB, and COMP are configured within the HAL drivers of these peripheral through the macros given in the table below. The EXTI internal connections depend on the targeted STM32 microcontroller (refer to the product datasheet for more details):

Table 13: Description of EXTI configuration macros

Macros	Description
PPP_EXTI_LINE_FUNCTION	Defines the EXTI line connected to the internal peripheral. Example: <code>#define PWR_EXTI_LINE_PVD ((uint32_t)0x00010000) /*!<External interrupt line 16 Connected to the PVD EXTI Line */</code>
_HAL_PPP_EXTI_ENABLE_IT	Enables a given EXTI line Example: <code>_HAL_PWR_PVD_EXTI_ENABLE_IT()</code>
_HAL_PPP_EXTI_DISABLE_IT	Disables a given EXTI line. Example: <code>_HAL_PWR_PVD_EXTI_DISABLE_IT()</code>
_HAL_PPP_EXTI_GET_FLAG	Gets a given EXTI line interrupt flag pending bit status. Example: <code>_HAL_PWR_PVD_EXTI_GET_FLAG()</code>

Macros	Description
<code>_HAL_PPP_EXTI_CLEAR_FLAG</code>	Clears a given EXTI line interrupt flag pending bit. Example: <code>_HAL_PWR_PVD_EXTI_CLEAR_FLAG()</code>
<code>_HAL_PPP_EXTI_GENERATE_SWIT</code>	Generates a software interrupt for a given EXTI line. Example: <code>_HAL_PWR_PVD_EXTI_GENERATE_SWIT()</code>
<code>_HAL_PPP_EXTI_ENABLE_EVENT</code>	Enables event on a given EXTI Line Example: <code>_HAL_PWR_PVD_EXTI_ENABLE_EVENT()</code>
<code>_HAL_PPP_EXTI_DISABLE_EVENT</code>	Disables event on a given EXTI line Example: <code>_HAL_PWR_PVD_EXTI_DISABLE_EVENT()</code>

If the EXTI interrupt mode is selected, the user application must call `HAL_PPP_FUNCTION_IRQHandler()` (for example `HAL_PWR_PVD_IRQHandler()`), from `stm32f3xx_it.c` file, and implement `HAL_PPP_FUNCTIONCallback()` callback function (for example `HAL_PWR_PVDCallback()`).

2.11.6 DMA

The DMA HAL driver allows enabling and configuring the peripheral to be connected to the DMA Channels (except for internal SRAM/FLASH memory which do not require any initialization). Refer to the product reference manual for details on the DMA request corresponding to each peripheral.

For a given channel, `HAL_DMA_Init()` API allows programming the required configuration through the following parameters:

- Transfer Direction
- Source and Destination data formats
- Circular, Normal or peripheral flow control mode
- Channels Priority level
- Source and Destination Increment mode
- FIFO mode and its Threshold (if needed)
- Burst mode for Source and/or Destination (if needed).

Two operating modes are available:

- Polling mode I/O operation
 - a. Use `HAL_DMA_Start()` to start DMA transfer when the source and destination addresses and the Length of data to be transferred have been configured.
 - b. Use `HAL_DMA_PollForTransfer()` to poll for the end of current transfer. In this case a fixed timeout can be configured depending on the user application.
- Interrupt mode I/O operation
 - a. Configure the DMA interrupt priority using `HAL_NVIC_SetPriority()`
 - b. Enable the DMA IRQ handler using `HAL_NVIC_EnableIRQ()`
 - c. Use `HAL_DMA_Start_IT()` to start DMA transfer when the source and destination addresses and the length of data to be transferred have been configured. In this case the DMA interrupt is configured.
 - d. Use `HAL_DMA_IRQHandler()` called under `DMA_IRQHandler()` Interrupt subroutine

- e. When data transfer is complete, HAL_DMA_IRQHandler() function is executed and a user function can be called by customizing XferCpltCallback and XferErrorCallback function pointer (i.e. a member of DMA handle structure).

Additional functions and macros are available to ensure efficient DMA management:

- Use HAL_DMA_GetState() function to return the DMA state and HAL_DMA_GetError() in case of error detection.
- Use HAL_DMA_Abort() function to abort the current transfer

The most used DMA HAL driver macros are the following:

- __HAL_DMA_ENABLE: enables the specified DMA Channels.
- __HAL_DMA_DISABLE: disables the specified DMA Channels.
- __HAL_DMA_GET_FLAG: gets the DMA Channels pending flags.
- __HAL_DMA_CLEAR_FLAG: clears the DMA Channels pending flags.
- __HAL_DMA_ENABLE_IT: enables the specified DMA Channels interrupts.
- __HAL_DMA_DISABLE_IT: disables the specified DMA Channels interrupts.
- __HAL_DMA_GET_IT_SOURCE: checks whether the specified DMA channel interrupt has occurred or not.



When a peripheral is used in DMA mode, the DMA initialization should be done in the HAL_PPP_MspInit() callback. In addition, the user application should associate the DMA handle to the PPP handle (refer to section “HAL IO operation functions”).



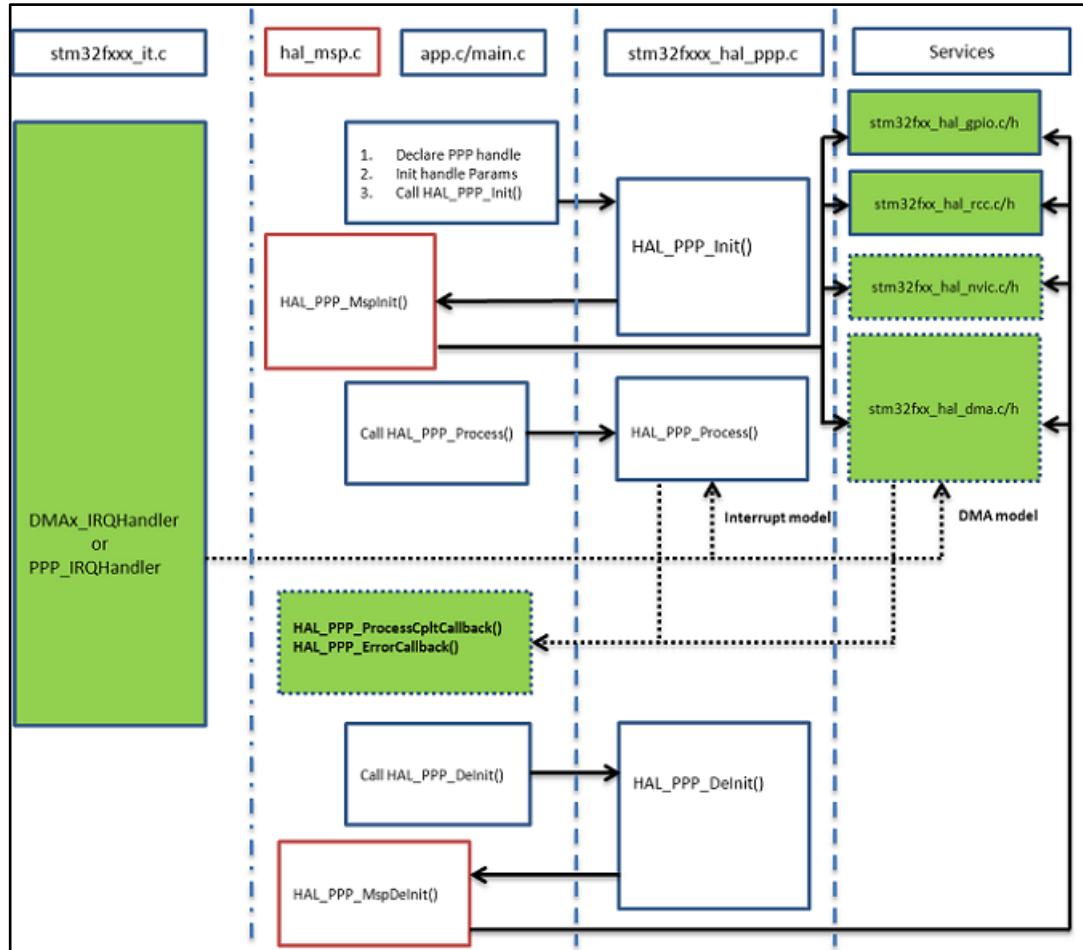
DMA channel callbacks need to be initialized by the user application only in case of memory-to-memory transfer. However when peripheral-to-memory transfers are used, these callbacks are automatically initialized by calling a process API function that uses the DMA.

2.12 How to use HAL drivers

2.12.1 HAL usage models

The following figure shows the typical use of the HAL driver and the interaction between the application user, the HAL driver and the interrupts.

Figure 7: HAL driver model



The functions implemented in the HAL driver are shown in green, the functions called from interrupt handlers in dotted lines, and the msp functions implemented in the user application in red. Non-dotted lines represent the interactions between the user application functions.

Basically, the HAL driver APIs are called from user files and optionally from interrupt handlers file when the APIs based on the DMA or the PPP peripheral dedicated interrupts are used.

When DMA or PPP peripheral interrupts are used, the PPP process complete callbacks are called to inform the user about the process completion in real-time event mode (interrupts). Note that the same process completion callbacks are used for DMA in interrupt mode.

2.12.2 HAL initialization

2.12.2.1 HAL global initialization

In addition to the peripheral initialization and de-initialization functions, a set of APIs are provided to initialize the HAL core implemented in file `stm32f3xx_hal.c`.

- `HAL_Init()`: this function must be called at application startup to
 - initialize data/instruction cache and pre-fetch queue
 - set Systick timer to generate an interrupt each 1ms (based on HSI clock) with the lowest priority
 - call `HAL_MspInit()` user callback function to perform system level initializations (Clock, GPIOs, DMA, interrupts). `HAL_MspInit()` is defined as “weak” empty function in the HAL drivers.
- `HAL_DeInit()`
 - resets all peripherals
 - calls function `HAL_MspDeInit()` which a is user callback function to do system level De-Initializations.
- `HAL_GetTick()`: this function gets current SysTick counter value (incremented in SysTick interrupt) used by peripherals drivers to handle timeouts.
- `HAL_Delay()`: this function implements a delay (expressed in milliseconds) using the SysTick timer.

Care must be taken when using `HAL_Delay()` since this function provides an accurate delay (expressed in milliseconds) based on a variable incremented in SysTick ISR. This means that if `HAL_Delay()` is called from a peripheral ISR, then the SysTick interrupt must have highest priority (numerically lower) than the peripheral interrupt, otherwise the caller ISR will be blocked.

2.12.2.2 System clock initialization

The clock configuration is done at the beginning of the user code. However the user can change the configuration of the clock in his own code. Please find below the typical Clock configuration sequence:

```
void SystemClock_Config(void)
{
  RCC_ClkInitTypeDef RCC_ClkInitStruct;
  RCC_OscInitTypeDef RCC_OscInitStruct;
  /* Enable HSE Oscillator and activate PLL with HSE as source */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.HSEPredivValue = RCC_HSE_PREDIV_DIV1;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
  RCC_OscInitStruct.PLL.PLLMUL = RCC_PLL_MUL9;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }
  /* Select PLL as system clock source and configure the HCLK, PCLK1 and PCLK2
  clocks dividers */
  RCC_ClkInitStruct.ClockType = (RCC_CLOCKTYPE_SYSCLK | RCC_CLOCKTYPE_HCLK |
  RCC_CLOCKTYPE_PCLK1 | RCC_CLOCKTYPE_PCLK2);
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV2;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;
  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
  {
    Error_Handler();
  }
}
```

2.12.2.3 HAL MSP initialization process

The peripheral initialization is done through *HAL_PPP_Init()* while the hardware resources initialization used by a peripheral (PPP) is performed during this initialization by calling MSP callback function *HAL_PPP_MspInit()*.

The *MspInit* callback performs the low level initialization related to the different additional hardware resources: RCC, GPIO, NVIC and DMA.

All the HAL drivers with handles include two MSP callbacks for initialization and de-initialization:

```
/** 
 * @brief Initializes the PPP MSP.
 * @param hppp: PPP handle
 * @retval None */
void __weak HAL_PPP_MspInit(PPP_HandleTypeDefDef *hppp) {
/* NOTE : This function Should not be modified, when the callback is needed,
the HAL PPP MspInit could be implemented in the user file */
}
/** 
 * @brief DeInitializes PPP MSP.
 * @param hppp: PPP handle
 * @retval None */
void __weak HAL_PPP_MspDeInit(PPP_HandleTypeDefDef *hppp) {
/* NOTE : This function Should not be modified, when the callback is needed,
the HAL PPP MspDeInit could be implemented in the user file */
}
```

The MSP callbacks are declared empty as weak functions in each peripheral driver. The user can use them to set the low level initialization code or omit them and use his own initialization routine.

The HAL MSP callback is implemented inside the *stm32f3xx_hal_msp.c* file in the user folders. An *stm32f3xx_hal_msp_template.c* file is located in the HAL folder and should be copied to the user folder. It can be generated automatically by STM32CubeMX tool and further modified. Note that all the routines are declared as weak functions and could be overwritten or removed to use user low level initialization code.

stm32f3xx_hal_msp.c file contains the following functions:

Table 14: MSP functions

Routine	Description
void HAL_MspInit()	Global MSP initialization routine
void HAL_MspDeInit()	Global MSP de-initialization routine
void HAL_PPP_MspInit()	PPP MSP initialization routine
void HAL_PPP_MspDeInit()	PPP MSP de-initialization routine

By default, if no peripheral needs to be de-initialized during the program execution, the whole MSP initialization is done in *Hal_MspInit()* and MSP De-Initialization in the *Hal_MspDeInit()*. In this case the *HAL_PPP_MspInit()* and *HAL_PPP_MspDeInit()* are not implemented.

When one or more peripherals needs to be de-initialized in run time and the low level resources of a given peripheral need to be released and used by another peripheral, *HAL_PPP_MspDeInit()* and *HAL_PPP_MspInit()* are implemented for the concerned peripheral and other peripherals initialization and de-Initialization are kept in the global *HAL_MspInit()* and the *HAL_MspDeInit()*.

If there is nothing to be initialized by the global *HAL_MspInit()* and *HAL_MspDeInit()*, the two routines can simply be omitted.

2.12.3 HAL IO operation process

The HAL functions with internal data processing like transmit, receive, write and read are generally provided with three data processing modes as follows:

- Polling mode
- Interrupt mode
- DMA mode

2.12.3.1 Polling mode

In Polling mode, the HAL functions return the process status when the data processing in blocking mode is complete. The operation is considered complete when the function returns the HAL_OK status, otherwise an error status is returned. The user can get more information through the *HAL_PPP_GetState()* function. The data processing is handled internally in a loop. A timeout (expressed in ms) is used to prevent process hanging.

The example below shows the typical Polling mode processing sequence :

```
HAL_StatusTypeDef HAL_PPP_Transmit ( PPP_HandleTypeDef * phandle, uint8_t pData,
int16_t tSize, uint32_t tTimeout)
{
if((pData == NULL ) || (Size == 0))
{
return HAL_ERROR;
}
(...) while (data processing is running)
{
if( timeout reached )
{
return HAL_TIMEOUT;
}
}
(...)
return HAL_OK; }
```

2.12.3.2 Interrupt mode

In Interrupt mode, the HAL function returns the process status after starting the data processing and enabling the appropriate interruption. The end of the operation is indicated by a callback declared as a weak function. It can be customized by the user to be informed in real-time about the process completion. The user can also get the process status through the *HAL_PPP_GetState()* function.

In Interrupt mode, four functions are declared in the driver:

- *HAL_PPP_Process_IT()*: launch the process
- *HAL_PPP_IRQHandler()*: the global PPP peripheral interruption
- *__weak HAL_PPP_ProcessCpltCallback ()*: the callback relative to the process completion.
- *__weak HAL_PPP_ProcessErrorCallback()*: the callback relative to the process Error.

To use a process in Interrupt mode, *HAL_PPP_Process_IT()* is called in the user file and *HAL_PPP_IRQHandler* in *stm32f3xx_it.c*.

The *HAL_PPP_ProcessCpltCallback()* function is declared as weak function in the driver. This means that the user can declare it again in the application. The function in the driver is not modified.

An example of use is illustrated below:

main.c file:

```
UART_HandleTypeDef UartHandle;
int main(void)
{
/* Set User Parameters */
UartHandle.Init.BaudRate = 9600;
UartHandle.Init.WordLength = UART_DATABITS_8;
UartHandle.Init.StopBits = UART_STOPBITS_1;
UartHandle.Init.Parity = UART_PARITY_NONE;
UartHandle.Init.HwFlowCtl = UART_HWCONTROL_NONE;
UartHandle.Init.Mode = UART_MODE_TX_RX;
UartHandle.Init.Instance = USART1;
HAL_UART_Init(&UartHandle);
HAL_UART_SendIT(&UartHandle, TxBuffer, sizeof(TxBuffer));
while (1);
}
void HAL_UART_TxCpltCallback(UART_HandleTypeDef *huart)
{
}
void HAL_UART_ErrorCallback(UART_HandleTypeDef *huart)
{}
```

stm32f3xx_it.c file:

```
extern UART_HandleTypeDef UartHandle;
void USART1_IRQHandler(void)
{
HAL_UART_IRQHandler(&UartHandle);
}
```

2.12.3.3 DMA mode

In DMA mode, the HAL function returns the process status after starting the data processing through the DMA and after enabling the appropriate DMA interruption. The end of the operation is indicated by a callback declared as a weak function and can be customized by the user to be informed in real-time about the process completion. The user can also get the process status through the *HAL_PPP_GetState()* function. For the DMA mode, three functions are declared in the driver:

- *HAL_PPP_Process_DMA()*: launch the process
- *HAL_PPP_DMA_IRQHandler()*: the DMA interruption used by the PPP peripheral
- *__weak HAL_PPP_ProcessCpltCallback()*: the callback relative to the process completion.
- *__weak HAL_PPP_ErrorCpltCallback()*: the callback relative to the process Error.

To use a process in DMA mode, *HAL_PPP_Process_DMA()* is called in the user file and the *HAL_PPP_DMA_IRQHandler()* is placed in the *stm32f3xx_it.c*. When DMA mode is used, the DMA initialization is done in the *HAL_PPP_MspInit()* callback. The user should also associate the DMA handle to the PPP handle. For this purpose, the handles of all the peripheral drivers that use the DMA must be declared as follows:

```
typedef struct
{
PPP_TypeDef *Instance; /* Register base address */
PPP_InitTypeDef Init; /* PPP communication parameters */
HAL_StateTypeDef State; /* PPP communication state */
(...)

DMA_HandleTypeDef *hdma; /* associated DMA handle */
} PPP_HandleTypeDef;
```

The initialization is done as follows (UART example):

```
int main(void)
{
/* Set User Parameters */
UartHandle.Init.BaudRate = 9600;
UartHandle.Init.WordLength = UART_DATABITS_8;
UartHandle.Init.StopBits = UART_STOPBITS_1;
UartHandle.Init.Parity = UART_PARITY_NONE;
UartHandle.Init.HwFlowCtl = UART_HWCONTROL_NONE;
UartHandle.Init.Mode = UART_MODE_TX_RX;
UartHandle.Init.Instance = USART1;
HAL_UART_Init(&UartHandle);
(...)
```

```
void HAL_USART_MspInit(UART_HandleTypeDef *huart)
{
static DMA_HandleTypeDef hdma_tx;
static DMA_HandleTypeDef hdma_rx;
(...)
__HAL_LINKDMA(UartHandle, DMA_Handle_tx, hdma_tx);
__HAL_LINKDMA(UartHandle, DMA_Handle_rx, hdma_rx);
(...)
```

}

The *HAL_PPP_ProcessCpltCallback()* function is declared as weak function in the driver that means, the user can declare it again in the application code. The function in the driver should not be modified.

An example of use is illustrated below:

main.c file:

```
UART_HandleTypeDef UartHandle;
int main(void)
{
/* Set User Parameters */
UartHandle.Init.BaudRate = 9600;
UartHandle.Init.WordLength = UART_DATABITS_8;
UartHandle.Init.StopBits = UART_STOPBITS_1;
UartHandle.Init.Parity = UART_PARITY_NONE;
UartHandle.Init.HwFlowCtl = UART_HWCONTROL_NONE;
UartHandle.Init.Mode = UART_MODE_TX_RX; UartHandle.Init.Instance = USART1;
HAL_UART_Init(&UartHandle);
HAL_UART_Send_DMA(&UartHandle, TxBuffer, sizeof(TxBuffer));
while (1);
```

```
void HAL_UART_TxCpltCallback(UART_HandleTypeDef *phuart)
{}
```

```
void HAL_UART_TxErrorCallback(UART_HandleTypeDef *phuart)
{}
```

stm32f3xx_it.c file:

```
extern UART_HandleTypeDef UartHandle;
void DMAx_IRQHandler(void)
{
HAL_DMA_IRQHandler(&UartHandle.DMA_HandleTypeDef_tx);
```

HAL_USART_TxCpltCallback() and *HAL_USART_ErrorCallback()* should be linked in the *HAL_PPP_Process_DMA()* function to the DMA transfer complete callback and the DMA transfer Error callback by using the following statement:

```
HAL_PPP_Process_DMA (PPP_HandleTypeDef *hppp, Params...)
{
(...)
```

```
hppp->DMA_HandleTypeDef->XferCpltCallback = HAL_UART_TxCpltCallback ;
hppp->DMA_HandleTypeDef->XferErrorCallback = HAL_UART_ErrorCallback ;
```

```
(...)
}
```

2.12.4 Timeout and error management

2.12.4.1 Timeout management

The timeout is often used for the APIs that operate in polling mode. It defines the delay during which a blocking process should wait till an error is returned. An example is provided below:

```
HAL_StatusTypeDef HAL_DMA_PollForTransfer(DMA_HandleTypeDef *hdma, uint32_t CompleteLevel, uint32_t Timeout)
```

The timeout possible value are the following:

Table 15: Timeout values

Timeout value	Description
0	No poll : Immediate process check and exit
1 ... (HAL_MAX_DELAY -1) ⁽¹⁾	Timeout in ms
HAL_MAX_DELAY	Infinite poll till process is successful

Notes:

⁽¹⁾HAL_MAX_DELAY is defined in the stm32fxx_hal_def.h as 0xFFFFFFFF

However, in some cases, a fixed timeout is used for system peripherals or internal HAL driver processes. In these cases, the timeout has the same meaning and is used in the same way, except when it is defined locally in the drivers and cannot be modified or introduced as an argument in the user application.

Example of fixed timeout:

```
#define LOCAL_PROCESS_TIMEOUT 100
HAL_StatusTypeDef HAL_PPP_Process(PPP_HandleTypeDef)
{
(
...
timeout = HAL_GetTick() + LOCAL_PROCESS_TIMEOUT;
(
...
while(ProcessOngoing)
{
(
...
if(HAL_GetTick() >= timeout)
{
/* Process unlocked */
HAL_UNLOCK(hppp);
hppp->State= HAL_PPP_STATE_TIMEOUT;
return HAL_PPP_STATE_TIMEOUT;
}
(
...
}
```

The following example shows how to use the timeout inside the polling functions:

```
HAL_PPP_StateTypeDef HAL_PPP_Poll (PPP_HandleTypeDef *hppp, uint32_t Timeout)
{
(
...
timeout = HAL_GetTick() + Timeout;
(
...
while(ProcessOngoing)
{
(
...
if(Timeout != HAL_MAX_DELAY)
{
```

```

if(HAL_GetTick() >= timeout)
{
/* Process unlocked */
__HAL_UNLOCK(hppp);
hppp->State= HAL PPP STATE TIMEOUT;
return hppp->State;
}
}
(...)
```

2.12.4.2 Error management

The HAL drivers implement a check on the following items:

- Valid parameters: for some process the used parameters should be valid and already defined, otherwise the system may crash or go into an undefined state. These critical parameters are checked before being used (see example below).

```

HAL_StatusTypeDef HAL_PPP_Process(PPP_HandleTypeDef* hppp, uint32_t *pdata, uint32
Size)
{ if ((pData == NULL) || (Size == 0))
{ return HAL_ERROR;
}
```

- Valid handle: the PPP peripheral handle is the most important argument since it keeps the PPP driver vital parameters. It is always checked in the beginning of the *HAL_PPP_Init()* function.

```

HAL_StatusTypeDef HAL_PPP_Init(PPP_HandleTypeDef* hppp)
{ if (hppp == NULL) /*the handle should be already allocated
{ return HAL_ERROR;
}
```

- Timeout error: the following statement is used when a timeout error occurs:

```

while (Process ongoing)
{ timeout = HAL_GetTick() + Timeout;
  while (data processing is running)
{ if(timeout)
{ return HAL_TIMEOUT;
}
}
```

When an error occurs during a peripheral process, *HAL_PPP_Process()* returns with a *HAL_ERROR* status. The HAL PPP driver implements the *HAL_PPP_GetError()* to allow retrieving the origin of the error.

```
HAL_PPP_ErrorTypeDef HAL_PPP_GetError (PPP_HandleTypeDef *hppp);
```

In all peripheral handles, a *HAL_PPP_ErrorTypeDef* is defined and used to store the last error code.

```

typedef struct
{
PPP_TypeDef * Instance; /* PPP registers base address */
PPP_InitTypeDef Init; /* PPP initialization parameters */
HAL_LockTypeDef Lock; /* PPP locking object */
__IO HAL_PPP_StateTypeDef State; /* PPP state */
__IO HAL_PPP_ErrorTypeDef ErrorCode; /* PPP Error code */
(...)
/* PPP specific parameters */
}
PPP_HandleTypeDef;
```

The error state and the peripheral global state are always updated before returning an error:

```
PPP->State = HAL_PPP_READY; /* Set the peripheral ready */
PPP->ErrorCode = HAL_ERRORCODE ; /* Set the error code */
HAL_UNLOCK(PPP) ; /* Unlock the PPP resources */
return HAL_ERROR; /*return with HAL error */
```

HAL_PPP_GetError () must be used in interrupt mode in the error callback:

```
void HAL_PPP_ProcessCpltCallback(PPP_HandleTypeDef *hspi)
{
    ErrorCode = HAL_PPP_GetError (hppp); /* retreive error code */
}
```

2.12.4.3 Run-time checking

The HAL implements run-time failure detection by checking the input values of all HAL driver functions. The run-time checking is achieved by using an *assert_param* macro. This macro is used in all the HAL drivers' functions which have an input parameter. It allows verifying that the input value lies within the parameter allowed values.

To enable the run-time checking, use the *assert_param* macro, and leave the define **USE_FULL_ASSERT** uncommented in *stm32f3xx_hal_conf.h* file.

```
void HAL_UART_Init(UART_HandleTypeDef *huart)
{
    (...) /* Check the parameters */
    assert_param(IS_UART_INSTANCE(huart->Instance));
    assert_param(IS_UART_BAUDRATE(huart->Init.BaudRate));
    assert_param(IS_UART_WORD_LENGTH(huart->Init.WordLength));
    assert_param(IS_UART_STOPBITS(huart->Init.StopBits));
    assert_param(IS_UART_PARITY(huart->Init.Parity));
    assert_param(IS_UART_MODE(huart->Init.Mode));
    assert_param(IS_UART_HARDWARE_FLOW_CONTROL(huart->Init.HwFlowCtl));
    (...)

    /** @defgroup UART_Word_Length *
     @{
     */
#define UART_WORDLENGTH_8B ((uint32_t)0x00000000)
#define UART_WORDLENGTH_9B ((uint32_t)USART_CR1_M)
#define IS_UART_WORD_LENGTH(LENGTH) (((LENGTH) == UART_WORDLENGTH_8B) ||
\ ((LENGTH) == UART_WORDLENGTH_9B))
```

If the expression passed to the *assert_param* macro is false, the *assert_failed* function is called and returns the name of the source file and the source line number of the call that failed. If the expression is true, no value is returned.

The *assert_param* macro is implemented in *stm32f3xx_hal_conf.h*:

```
/* Exported macro -----
#ifndef USE_FULL_ASSERT
/***
 * @brief The assert param macro is used for function's parameters check.
 * @param expr: If expr is false, it calls assert failed function
 * which reports the name of the source file and the source
 * line number of the call that failed.
 * If expr is true, it returns no value.
 * @retval None */
#define assert_param(expr) ((expr)?(void)0:assert_failed((uint8_t *) FILE ,
LINE))
/* Exported functions -----*/
void assert_failed(uint8_t * file, uint32_t line);
#else
#define assert_param(expr)((void)0)
#endif /* USE_FULL_ASSERT */
```

The `assert_failed` function is implemented in the main.c file or in any other user C file:

```
#ifdef USE_FULL_ASSERT /**
 * @brief Reports the name of the source file and the source line number
 * where the assert_param error has occurred.
 * @param file: pointer to the source file name
 * @param line: assert param error line source number
 * @retval None */
void assert_failed(uint8_t* file, uint32_t line)
{
/* User can add his own implementation to report the file name and line number,
ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
/* Infinite loop */
while (1)
{
}
```



Because of the overhead run-time checking introduces, it is recommended to use it during application code development and debugging, and to remove it from the final application to improve code size and speed.

3 Overview of low-layer drivers

The low-layer (LL) drivers are designed to offer a fast light-weight expert-oriented layer which is closer to the hardware than the HAL. Contrary to the HAL, LL APIs are not provided for peripherals where optimized access is not a key feature, or those requiring heavy software configuration and/or complex upper-level stack (such as FSMC or USB).

The LL drivers feature:

- A set of functions to initialize peripheral main features according to the parameters specified in data structures
- A set of functions used to fill initialization data structures with the reset values of each field
- Functions to perform peripheral de-initialization (peripheral registers restored to their default values)
- A set of inline functions for direct and atomic register access
- Full independence from HAL since LL drivers can be used either in standalone mode (without HAL drivers) or in mixed mode (with HAL drivers)
- Full coverage of the supported peripheral features.

The low-layer drivers provide hardware services based on the available features of the STM32 peripherals. These services reflect exactly the hardware capabilities and provide one-shot operations that must be called following the programming model described in the microcontroller line reference manual. As a result, the LL services do not implement any processing and do not require any additional memory resources to save their states, counter or data pointers: all the operations are performed by changing the associated peripheral registers content.

3.1 Low-layer files

The low-layer drivers are built around header/C files (one per each supported peripheral) plus five header files for some System and Cortex related features.

Table 16: LL driver files

File	Description
<i>stm32f3xx_ll_bus.h</i>	This is the h-source file for core bus control and peripheral clock activation and deactivation <i>Example: LL_AHB1_GRP1_EnableClock</i>
<i>stm32f3xx_ll_ppp.h/c</i>	<i>stm32f3xx_ll_ppp.c</i> provides peripheral initialization functions such as LL_PPP_Init(), LL_PPP_StructInit(), LL_PPP_DelInit(). All the other APIs are defined within <i>stm32f3xx_ll_ppp.h</i> file. The low-layer PPP driver is a standalone module. To use it, the application must include it in the <i>xx_ll_ppp.h</i> file.
<i>stm32xx_ll_cortex.h</i>	CortexM related register operation APIs including the Systick, Low power (LL_SYSTICK_xxxxx, LL_LPM_xxxxx "Low Power Mode" ...)
<i>stm32xx_ll_xx_ll_utils.h/c</i>	This file covers the generic APIs: <ul style="list-style-type: none"> • Read of device unique ID and electronic signature • Timebase and delay management • System clock configuration.
<i>stm32xx_ll_xx_ll_system.h</i>	System related operations (LL_SYSCFG_xxx, LL_DBGMCU_xxx, LL_FLASH_xxx)

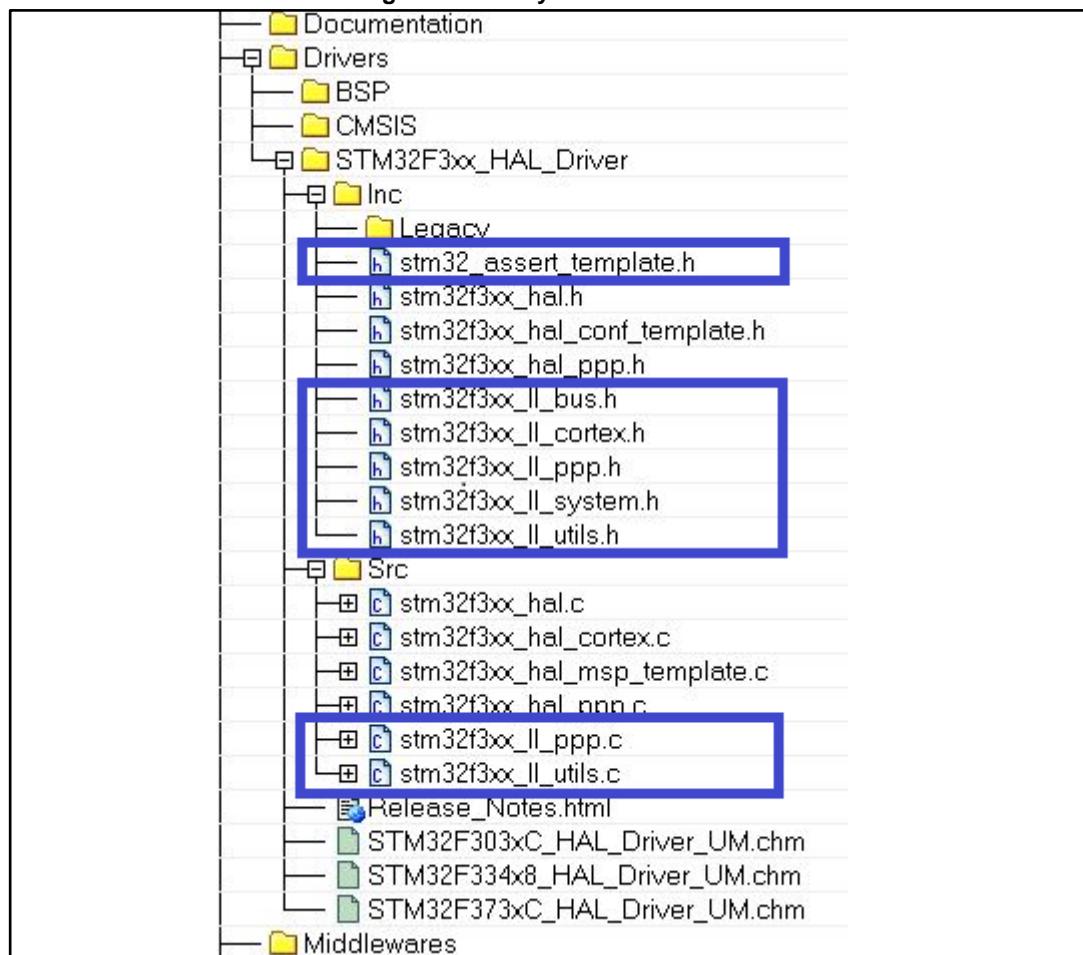
File	Description
stm32_assert_template.h	Template file allowing to define the assert_param macro, that is used when run-time checking is enabled. This file is required only when the LL drivers are used in standalone mode (without calling the HAL APIs). It should be copied to the application folder and renamed to stm32_assert.h.



There is no configuration file for the LL drivers.

The low-layer files are located in the same HAL driver folder.

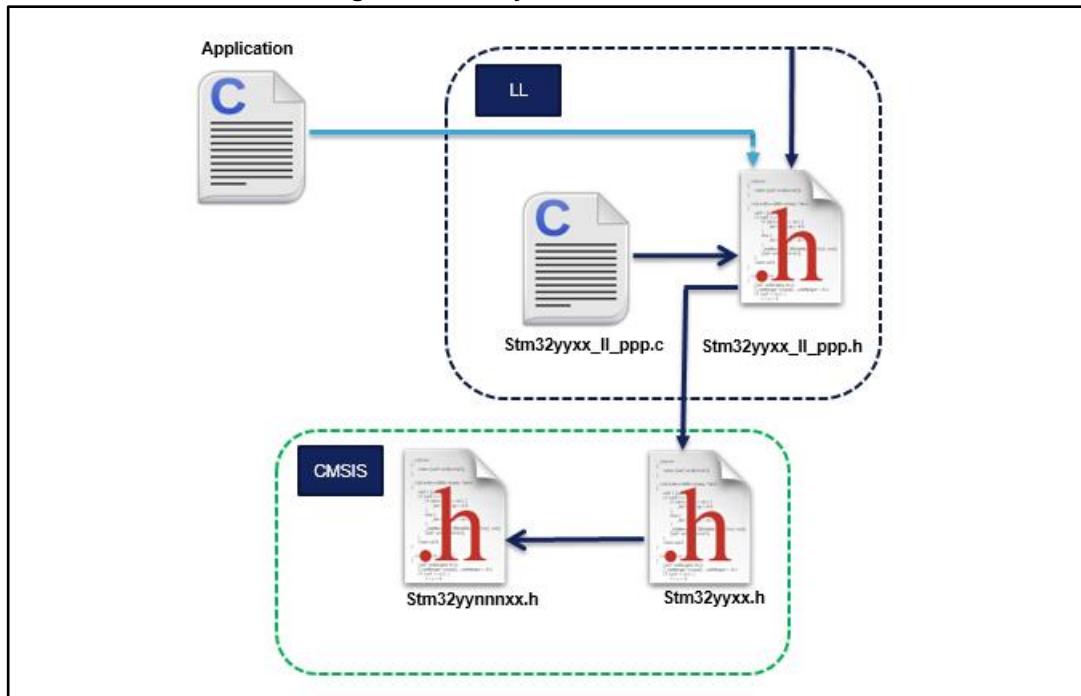
Figure 8: low-layer driver folders



In general, low-layer drivers include only the STM32 CMSIS device file.

```
#include "stm32yyxx.h"
```

Figure 9: Low-layer driver CMSIS files



Application files have to include only the used low-layer drivers header files.

3.2 Overview of low-layer APIs and naming rules

3.2.1 Peripheral initialization functions

The LL drivers offer three sets of initialization functions. They are defined in `stm32f3xx_ll_ppp.c` file:

- Functions to initialize peripheral main features according to the parameters specified in data structures
- A set of functions used to fill initialization data structures with the reset values of each field
- Function for peripheral de-initialization (peripheral registers restored to their default values)

The definition of these LL initialization functions and associated resources (structure, literals and prototypes) is conditioned by a compilation switch: `USE_FULL_LL_DRIVER`. To use these functions, this switch must be added in the toolchain compiler preprocessor or to any generic header file which is processed before the LL drivers.

The below table shows the list of the common functions provided for all the supported peripherals:

Table 17: Common peripheral initialization functions

Functions	Return Type	Parameters	Description
LL_PPP_Init	ErrorStatus	<ul style="list-style-type: none"> • <i>PPP_TypeDef* PPPx</i> • <i>LL_PPP_InitTypeDef* PPP_InitStruct</i> 	Initializes the peripheral main features according to the parameters specified in PPP_InitStruct. Example: LL_USART_Init(USART_TypeDef *USARTx, LL_USART_InitTypeDef *USART_InitStruct)
LL_PPP_StructInit	void	<ul style="list-style-type: none"> • <i>LL_PPP_InitTypeDef* PPP_InitStruct</i> 	Fills each PPP_InitStruct member with its default value. Example. LL_USART_StructInit(LL_USART_InitTypeDef *USART_InitStruct)
LL_PPP_DeInit	ErrorStatus	<ul style="list-style-type: none"> • <i>PPP_TypeDef* PPPx</i> 	De-initializes the peripheral registers, that is restore them to their default reset values. Example. LL_USART_DeInit(USART_TypeDef *USARTx)

Additional functions are available for some peripherals (refer to [Table 18: "Optional peripheral initialization functions"](#))

Table 18: Optional peripheral initialization functions

Functions	Return Type	Parameters	Examples
LL_PPP{CATEGORY}_Init	Error Status	<ul style="list-style-type: none"> • <i>PPP_TypeDef* PPPx</i> • <i>LL_PPP{CATEGORY}_InitTypeDef* PPP{CATEGORY}_InitStruct</i> 	<p>Initializes peripheral features according to the parameters specified in PPP_InitStruct.</p> <p>Example:</p> <p>LL_ADC_INJ_Init(ADC_TypeDef *ADCx, LL_ADC_INJ_InitTypeDef *ADC_INJ_InitStruct)</p> <p>LL_RTC_TIME_Init(RTC_TypeDef *RTCx, uint32_t RTC_Format, LL_RTC_TimeTypeDef *RTC_TimeStruct)</p> <p>LL_RTC_DATE_Init(RTC_TypeDef *RTCx, uint32_t RTC_Format, LL_RTC_DateTypeDef *RTC_DateStruct)</p> <p>LL_TIM_IC_Init(TIM_TypeDef* TIMx, uint32_t Channel, LL_TIM_IC_InitTypeDef* TIM_IC_InitStruct)</p> <p>LL_TIM_ENCODER_Init(TIM_TypeDef* TIMx, LL_TIM_ENCODER_InitTypeDef* TIM_EncoderInitStruct)</p>

Functions	Return Type	Parameters	Examples
LL_PPP_{CATEGORY}_StructInit	void	<i>LL_PPP_{CATEGORY}_InitTypeDef*</i> <i>PPP_{CATEGORY}_InitStruct</i>	Fills each <i>PPP_{CATEGORY}_InitStruct</i> member with its default value. Example: <code>LL_ADC_INJ_StructInit(LL_ADC_INJ_InitTypeDef *ADC_INJ_InitStruct)</code>
LL_PPP_CommonInit	Error Status	<ul style="list-style-type: none"> • <i>PPP_TypeDef* PPPx</i> • <i>LL_PPP_CommonInitTypeDef*</i> <i>PPP_CommonInitStruct</i> 	Initializes the common features shared between different instances of the same peripheral. Example: <code>LL_ADC_CommonInit(ADC_Common_TypeDef *ADCxy_COMMON, LL_ADC_CommonInitTypeDef *ADC_CommonInitStruct)</code>
LL_PPP_CommonStructInit	void	<i>LL_PPP_CommonInitTypeDef*</i> <i>PPP_CommonInitStruct</i>	Fills each <i>PPP_CommonInitStruct</i> member with its default value Example: <code>LL_ADC_CommonStructInit(LL_ADC_CommonInitTypeDef *ADC_CommonInitStruct)</code>
LL_PPP_ClockInit	Error Status	<ul style="list-style-type: none"> • <i>PPP_TypeDef* PPPx</i> • <i>LL_PPP_ClockInitTypeDef*</i> <i>PPP_ClockInitStruct</i> 	Initializes the peripheral clock configuration in synchronous mode. Example: <code>LL_USART_ClockInit(USART_TypeDef *USARTx, LL_USART_ClockInitTypeDef *USART_ClockInitStruct)</code>
LL_PPP_ClockStructInit	void	<i>LL_PPP_ClockInitTypeDef*</i> <i>PPP_ClockInitStruct</i>	Fills each <i>PPP_ClockInitStruct</i> member with its default value Example: <code>LL_USART_ClockStructInit(LL_USART_ClockInitTypeDef *USART_ClockInitStruct)</code>

3.2.1.1 Run-time checking

Like HAL drivers, LL initialization functions implement run-time failure detection by checking the input values of all LL driver functions. For more details please refer to [Section 2.12.4.3: "Run-time checking"](#).

When using the LL drivers in standalone mode (without calling HAL functions), the following actions are required to use run-time checking:

1. Copy `stm32_assert_template.h` to the application folder and rename it to `stm32_assert.h`. This file defines the `assert_param` macro which is used when run-time checking is enabled.
2. Include `stm32_assert.h` file within the application main header file.
3. Add the `USE_FULL_ASSERT` compilation switch in the toolchain compiler preprocessor or in any generic header file which is processed before the `stm32_assert.h` driver.



Run-time checking is not available for LL inline functions.

3.2.2 Peripheral register-level configuration functions

On top of the peripheral initialization functions, the LL drivers offer a set of inline functions for direct atomic register access. Their format is as follows:

```
_STATIC_INLINE return_type LL_PPP_Function (PPPx_TypeDef *PPPx, args)
```

The “Function” naming is defined depending to the action category:

- **Specific Interrupt, DMA request and status flags management:**
Set/Get/Clear/Enable/Disable flags on interrupt and status registers

Table 19: Specific Interrupt, DMA request and status flags management

Name	Examples
<code>LL_PPP_{_CATEGORY}_ActionItem_BITNAME</code>	<ul style="list-style-type: none"> • <code>LL_RCC_IsActiveFlag_LSIRDY</code> • <code>LL_RCC_IsActiveFlag_FWRST()</code> • <code>LL_ADC_ClearFlag_EOC(ADC1)</code> • <code>LL_DMA_ClearFlag_TCx(DMA_TypeDef* DMAx)</code>
<code>LL_PPP{_CATEGORY}_IsItem_BITNAME_Action</code>	

Table 20: Available function formats

Item	Action	Format
Flag	Get	<code>LL_PPP_IsActiveFlag_BITNAME</code>
	Clear	<code>LL_PPP_ClearFlag_BITNAME</code>
Interrupts	Enable	<code>LL_PPP_EnableIT_BITNAME</code>
	Disable	<code>LL_PPP_DisableIT_BITNAME</code>
	Get	<code>LL_PPP_IsEnabledIT_BITNAME</code>
DMA	Enable	<code>LL_PPP_EnableDMAReq_BITNAME</code>
	Disable	<code>LL_PPP_DisableDMAReq_BITNAME</code>
	Get	<code>LL_PPP_IsEnabledDMAReq_BITNAME</code>



BITNAME refers to the peripheral register bit name as described in the product line reference manual.

- **Peripheral clock activation/deactivation management:** Enable/Disable/Reset a peripheral clock

Table 21: Peripheral clock activation/deactivation management

Name	Examples
<code>LL_bus_GRPx_ActionClock{Mode}</code>	<ul style="list-style-type: none"> • <code>LL_AHB1_GRP1_EnableClock (LL_AHB1_GRP1_PERIPH_GPIOA LL_AHB1_GRP1_PERIPH_GPIOB)</code> • <code>LL_APB1_GRP1_EnableClockSleep (LL_APB1_GRP1_PERIPH_DAC1)</code>



'x' corresponds to the group index and refers to the index of the modified register on a given bus.



'bus' correspond to the bus name (for example APB1).

- **Peripheral activation/deactivation management:** Enable/disable a peripheral or activate/deactivate specific peripheral features

Table 22: Peripheral activation/deactivation management

Name	Examples
<code>LL_PPP{CATEGORY}_Action{Item}</code> <code>LL_PPP{CATEGORY}_IsItemAction</code>	<ul style="list-style-type: none"> • <code>LL_ADC_Enable()</code> • <code>LL_ADC_StartCalibration();</code> • <code>LL_ADC_IsCalibrationOnGoing();</code> • <code>LL_RCC_HSI_Enable()</code> • <code>LL_RCC_HSI_IsReady()</code>

- **Peripheral configuration management:** Set/get a peripheral configuration settings

Table 23: Peripheral configuration management

Name	Examples
<code>LL_PPP{CATEGORY}_Set{ or Get}ConfigItem</code>	<code>LL_USART_SetBaudRate(USART2, 16000000,</code> <code>LL_USART_OVERSAMPLING_16, 9600)</code>

- **Peripheral register management:** Write/read the content of a register/retrun DMA relative register address

Table 24: Peripheral register management

Name
<code>LL_PPP_WriteReg(__INSTANCE__, __REG__, __VALUE__)</code>
<code>LL_PPP_ReadReg(__INSTANCE__, __REG__)</code>
<code>LL_PPP_DMA_GetRegAddr(PPP_TypeDef *PPPx, {Sub Instance if any ex: Channel}, {uint32_t Proprietary})</code>



The Proprietary is a variable used to identify the DMA transfer direction or the data register type.

4 Cohabiting of HAL and LL

The low-layer APIs are designed to be used in standalone mode or combined with the HAL. They cannot be automatically used with the HAL for the same peripheral instance. If you use the LL APIs for a specific instance, you can still use the HAL APIs for other instances. Be careful that the low-layer APIs might overwrite some registers which content is mirrored in the HAL handles.

4.1 Low-layer driver used in standalone mode

The low-layer APIs can be used without calling the HAL driver services. This is done by simply including `stm32f3xx_ll_ppp.h` in the application files. The LL APIs for a given peripheral are called by executing the same sequence as the one recommended by the programming model in the corresponding product line reference manual. In this case the HAL drivers associated to the used peripheral can be removed from the workspace. However the STM32CubeF3 framework should be used in the same way as in the HAL drivers case which means that System file, startup file and CMSIS should always be used.



When the BSP drivers are included, the used HAL drivers associated with the BSP functions drivers should be included in the workspace, even if they are not used by the application layer.

4.2 Mixed use of low-layer APIs and HAL drivers

In this case the low-layer APIs are used in conjunction with the HAL drivers to achieve direct and register level based operations.

Mixed use is allowed, however some consideration should be taken into account:

- It is recommended to avoid using simultaneously the HAL APIs and the combination of low-layer APIs for a given peripheral instance. If this is the case, one or more private fields in the HAL PPP handle structure should be updated accordingly.
- For operations and processes that do not alter the handle fields including the initialization structure, the HAL driver APIs and the low-layer services can be used together for the same peripheral instance.
- The low-layer drivers can be used without any restriction with all the HAL drivers that are not based on handle objects (RCC, common HAL, flash and GPIO).

Several examples showing how to use HAL and LL in the same application are provided within STM32F3 firmware package (refer to Examples_MIX projects).



1. When the HAL Init/DeInit APIs are not used and are replaced by the low-layer macros, the `InitMsp()` functions are not called and the MSP initialization should be done in the user application.
2. When process APIs are not used and the corresponding function is performed through the low-layer APIs, the callbacks are not called and post processing or error management should be done by the user application.
3. When the LL APIs are used for process operations, the IRQ handler HAL APIs cannot be called and the IRQ should be implemented by the user application. Each LL driver implements the macros needed to read and clear the associated interrupt flags.

5 HAL System Driver

5.1 HAL Firmware driver API description

5.1.1 How to use this driver

The common HAL driver contains a set of generic and common APIs that can be used by the PPP peripheral drivers and the user to start using the HAL.

The HAL contains two APIs categories:

- HAL Initialization and de-initialization functions
- HAL Control functions

5.1.2 Initialization and de-initialization functions

This section provides functions allowing to:

- Initializes the Flash interface, the NVIC allocation and initial clock configuration. It initializes the source of time base also when timeout is needed and the backup domain when enabled.
- de-Initializes common part of the HAL.
- Configure The time base source to have 1ms time base with a dedicated Tick interrupt priority.
 - Systick timer is used by default as source of time base, but user can eventually implement his proper time base source (a general purpose timer for example or other time source), keeping in mind that Time base duration should be kept 1ms since PPP_TIMEOUT_VALUES are defined and handled in milliseconds basis.
 - Time base configuration function (HAL_InitTick ()) is called automatically at the beginning of the program after reset by HAL_Init() or at any time when clock is configured, by HAL_RCC_ClockConfig().
 - Source of time base is configured to generate interrupts at regular time intervals. Care must be taken if HAL_Delay() is called from a peripheral ISR process, the Tick interrupt line must have higher priority (numerically lower) than the peripheral interrupt. Otherwise the caller ISR process will be blocked.
 - functions affecting time base configurations are declared as __Weak to make override possible in case of other implementations in user file.

This section contains the following APIs:

- [**HAL_Init\(\)**](#)
- [**HAL_DeInit\(\)**](#)
- [**HAL_MspInit\(\)**](#)
- [**HAL_MspDeInit\(\)**](#)
- [**HAL_InitTick\(\)**](#)

5.1.3 HAL Control functions

This section provides functions allowing to:

- Provide a tick value in millisecond
- Provide a blocking delay in millisecond
- Suspend the time base source interrupt
- Resume the time base source interrupt
- Get the HAL API driver version
- Get the device identifier

- Get the device revision identifier
- Enable/Disable Debug module during Sleep mode
- Enable/Disable Debug module during STOP mode
- Enable/Disable Debug module during STANDBY mode

This section contains the following APIs:

- *[HAL_IncTick\(\)](#)*
- *[HAL_GetTick\(\)](#)*
- *[HAL_Delay\(\)](#)*
- *[HAL_SuspendTick\(\)](#)*
- *[HAL_ResumeTick\(\)](#)*
- *[HAL_GetHalVersion\(\)](#)*
- *[HAL_GetREVID\(\)](#)*
- *[HAL_GetDEVID\(\)](#)*
- *[HAL_GetUIDw0\(\)](#)*
- *[HAL_GetUIDw1\(\)](#)*
- *[HAL_GetUIDw2\(\)](#)*
- *[HAL_DBGMCU_EnableDBGSleepMode\(\)](#)*
- *[HAL_DBGMCU_DisableDBGSleepMode\(\)](#)*
- *[HAL_DBGMCU_EnableDBGStopMode\(\)](#)*
- *[HAL_DBGMCU_DisableDBGStopMode\(\)](#)*
- *[HAL_DBGMCU_EnableDBGStandbyMode\(\)](#)*
- *[HAL_DBGMCU_DisableDBGStandbyMode\(\)](#)*

5.1.4 Detailed description of functions

HAL_Init

Function name	HAL_StatusTypeDef HAL_Init (void)
Function description	This function configures the Flash prefetch, Configures time base source, NVIC and Low level hardware.
Return values	<ul style="list-style-type: none"> • HAL: status
Notes	<ul style="list-style-type: none"> • This function is called at the beginning of program after reset and before the clock configuration • The Systick configuration is based on HSI clock, as HSI is the clock used after a system Reset and the NVIC configuration is set to Priority group 4 • The time base configuration is based on MSI clock when exiting from Reset. Once done, time base tick start incrementing. In the default implementation,Systick is used as source of time base. The tick variable is incremented each 1ms in its ISR.

HAL_DeInit

Function name	HAL_StatusTypeDef HAL_DeInit (void)
Function description	This function de-Initializes common part of the HAL and stops the source of time base.
Return values	<ul style="list-style-type: none"> • HAL: status
Notes	<ul style="list-style-type: none"> • This function is optional.

HAL_MspInit

Function name **void HAL_MspInit (void)**

Function description Initializes the MSP.

Return values • **None**

HAL_MspDeInit

Function name **void HAL_MspDeInit (void)**

Function description DeInitializes the MSP.

Return values • **None**

HAL_InitTick

Function name **HAL_StatusTypeDef HAL_InitTick (uint32_t TickPriority)**

Function description This function configures the source of the time base.

- **TickPriority:** Tick interrupt priority.
- **HAL:** status
- This function is called automatically at the beginning of program after reset by HAL_Init() or at any time when clock is reconfigured by HAL_RCC_ClockConfig().
- In the default implementation , SysTick timer is the source of time base. It is used to generate interrupts at regular time intervals. Care must be taken if HAL_Delay() is called from a peripheral ISR process, The the SysTick interrupt must have higher priority (numerically lower) than the peripheral interrupt. Otherwise the caller ISR process will be blocked. The function is declared as __Weak to be overwritten in case of other implementation in user file.

HAL_IncTick

Function name **void HAL_IncTick (void)**

Function description This function is called to increment a global variable "uwTick" used as application time base.

Return values • **None**

- Notes
- In the default implementation, this variable is incremented each 1ms in Systick ISR.
 - This function is declared as __weak to be overwritten in case of other implementations in user file.

HAL_Delay

Function name **void HAL_Delay (__IO uint32_t Delay)**

Function description This function provides accurate delay (in milliseconds) based on variable incremented.

Parameters • **Delay:** specifies the delay time length, in milliseconds.

Return values	None
Notes	<ul style="list-style-type: none"> In the default implementation , SysTick timer is the source of time base. It is used to generate interrupts at regular time intervals where uwTick is incremented. The function is declared as __Weak to be overwritten in case of other implementations in user file.

HAL_SuspendTick

Function name	void HAL_SuspendTick (void)
Function description	Suspend Tick increment.
Return values	None
Notes	<ul style="list-style-type: none"> In the default implementation , SysTick timer is the source of time base. It is used to generate interrupts at regular time intervals. Once HAL_SuspendTick() is called, the the SysTick interrupt will be disabled and so Tick increment is suspended. This function is declared as __weak to be overwritten in case of other implementations in user file.

HAL_ResumeTick

Function name	void HAL_ResumeTick (void)
Function description	Resume Tick increment.
Return values	None
Notes	<ul style="list-style-type: none"> In the default implementation , SysTick timer is the source of time base. It is used to generate interrupts at regular time intervals. Once HAL_ResumeTick() is called, the the SysTick interrupt will be enabled and so Tick increment is resumed. The function is declared as __Weak to be overwritten in case of other implementations in user file.

HAL_GetTick

Function name	uint32_t HAL_GetTick (void)
Function description	Povides a tick value in millisecond.
Return values	tick: value
Notes	<ul style="list-style-type: none"> The function is declared as __Weak to be overwritten in case of other implementations in user file.

HAL_GetHalVersion

Function name	uint32_t HAL_GetHalVersion (void)
Function description	This function returns the HAL revision.
Return values	<ul style="list-style-type: none"> version: : 0xXYZR (8bits for each decimal, R for RC)

HAL_GetREVID

Function name	uint32_t HAL_GetREVID (void)
Function description	Returns the device revision identifier.
Return values	<ul style="list-style-type: none">• Device: revision identifier

HAL_GetDEVID

Function name	uint32_t HAL_GetDEVID (void)
Function description	Returns the device identifier.
Return values	<ul style="list-style-type: none">• Device: identifier

HAL_GetUIDw0

Function name	uint32_t HAL_GetUIDw0 (void)
Function description	Returns first word of the unique device identifier (UID based on 96 bits)
Return values	<ul style="list-style-type: none">• Device: identifier

HAL_GetUIDw1

Function name	uint32_t HAL_GetUIDw1 (void)
Function description	Returns second word of the unique device identifier (UID based on 96 bits)
Return values	<ul style="list-style-type: none">• Device: identifier

HAL_GetUIDw2

Function name	uint32_t HAL_GetUIDw2 (void)
Function description	Returns third word of the unique device identifier (UID based on 96 bits)
Return values	<ul style="list-style-type: none">• Device: identifier

HAL_DBGMCU_EnableDBGSleepMode

Function name	void HAL_DBGMCU_EnableDBGSleepMode (void)
Function description	Enable the Debug Module during SLEEP mode.
Return values	<ul style="list-style-type: none">• None

HAL_DBGMCU_DisableDBGSleepMode

Function name	void HAL_DBGMCU_DisableDBGSleepMode (void)
Function description	Disable the Debug Module during SLEEP mode.
Return values	<ul style="list-style-type: none">• None

HAL_DBGMCU_EnableDBGStopMode

Function name **void HAL_DBGMCU_EnableDBGStopMode (void)**

Function description Enable the Debug Module during STOP mode.

Return values • **None**

HAL_DBGMCU_DisableDBGStopMode

Function name **void HAL_DBGMCU_DisableDBGStopMode (void)**

Function description Disable the Debug Module during STOP mode.

Return values • **None**

HAL_DBGMCU_EnableDBGStandbyMode

Function name **void HAL_DBGMCU_EnableDBGStandbyMode (void)**

Function description Enable the Debug Module during STANDBY mode.

Return values • **None**

HAL_DBGMCU_DisableDBGStandbyMode

Function name **void HAL_DBGMCU_DisableDBGStandbyMode (void)**

Function description Disable the Debug Module during STANDBY mode.

Return values • **None**

5.2 HAL Firmware driver defines

5.2.1 HAL

CRC aliases for Exported Functions

HAL_CRC_Input_Data_Reverse

HAL_CRC_Output_Data_Reverse

HAL DMA Remapping

HAL_REMAPDMA_ADC24_DMA2_CH34

ADC24 DMA remap
(STM32F303xB/C/E,
STM32F358xx and STM32F398xx
devices) 1: Remap (ADC24 DMA
requests mapped on DMA2
channels 3 and 4)

HAL_REMAPDMA_TIM16_DMA1_CH6

TIM16 DMA request remap 1:
Remap (TIM16_CH1 and
TIM16_UP DMA requests mapped
on DMA1 channel 6)

HAL_REMAPDMA_TIM17_DMA1_CH7

TIM17 DMA request remap 1:
Remap (TIM17_CH1 and
TIM17_UP DMA requests mapped
on DMA1 channel 7)

HAL_REMAPDMA_TIM6_DAC1_CH1_DMA1_CH3	TIM6 and DAC channel1 DMA remap (STM32F303xB/C/E, STM32F358xx and STM32F398xx devices) 1: Remap (TIM6_UP and DAC_CH1 DMA requests mapped on DMA1 channel 3)
HAL_REMAPDMA_TIM7_DAC1_CH2_DMA1_CH4	TIM7 and DAC channel2 DMA remap (STM32F303xB/C/E, STM32F358xx and STM32F398xx devices) 1: Remap (TIM7_UP and DAC_CH2 DMA requests mapped on DMA1 channel 4)
HAL_REMAPDMA_DAC2_CH1_DMA1_CH5	DAC2 channel1 DMA remap (STM32F303x4/6/8 devices only) 1: Remap (DAC2_CH1 DMA requests mapped on DMA1 channel 5)
HAL_REMAPDMA_TIM18_DAC2_CH1_DMA1_CH5	DAC2 channel1 DMA remap (STM32F303x4/6/8 devices only) 1: Remap (DAC2_CH1 DMA requests mapped on DMA1 channel 5)

IS_DMA_REMAP***HAL CCM RAM page write protection***

HAL_SYSCFG_WP_PAGE0	ICODE SRAM Write protection page 0
HAL_SYSCFG_WP_PAGE1	ICODE SRAM Write protection page 1
HAL_SYSCFG_WP_PAGE2	ICODE SRAM Write protection page 2
HAL_SYSCFG_WP_PAGE3	ICODE SRAM Write protection page 3
HAL_SYSCFG_WP_PAGE4	ICODE SRAM Write protection page 4
HAL_SYSCFG_WP_PAGE5	ICODE SRAM Write protection page 5
HAL_SYSCFG_WP_PAGE6	ICODE SRAM Write protection page 6
HAL_SYSCFG_WP_PAGE7	ICODE SRAM Write protection page 7

IS_HAL_SYSCFG_WP_PAGE***HAL state definition***

HAL_SMBUS_STATE_RESET	SMBUS not yet initialized or disabled
HAL_SMBUS_STATE_READY	SMBUS initialized and ready for use
HAL_SMBUS_STATE_BUSY	SMBUS internal process is ongoing
HAL_SMBUS_STATE_MASTER_BUSY_TX	Master Data Transmission process is ongoing
HAL_SMBUS_STATE_MASTER_BUSY_RX	Master Data Reception process is ongoing
HAL_SMBUS_STATE_SLAVE_BUSY_TX	Slave Data Transmission process is ongoing
HAL_SMBUS_STATE_SLAVE_BUSY_RX	Slave Data Reception process is ongoing
HAL_SMBUS_STATE_TIMEOUT	Timeout state
HAL_SMBUS_STATE_ERROR	Reception process is ongoing

<code>HAL_SMBUS_STATE_LISTEN</code>	Address Listen Mode is ongoing
<i>HAL SYSCFG Interrupts</i>	
<code>HAL_SYSCFG_IT_FPU_IOC</code>	Floating Point Unit Invalid operation Interrupt
<code>HAL_SYSCFG_IT_FPU_DZC</code>	Floating Point Unit Divide-by-zero Interrupt
<code>HAL_SYSCFG_IT_FPU_UFC</code>	Floating Point Unit Underflow Interrupt
<code>HAL_SYSCFG_IT_FPU_OFC</code>	Floating Point Unit Overflow Interrupt
<code>HAL_SYSCFG_IT_FPU_IDC</code>	Floating Point Unit Input denormal Interrupt
<code>HAL_SYSCFG_IT_FPU_IXC</code>	Floating Point Unit Inexact Interrupt
<code>IS_HAL_SYSCFG_INTERRUPT</code>	
<i>HAL Trigger Remapping</i>	
<code>HAL_REMAPTRIGGER_DAC1_TRIG</code>	DAC trigger remap (when TSEL = 001 on STM32F303xB/C and STM32F358xx devices) 0: No remap (DAC trigger is TIM8_TRGO) 1: Remap (DAC trigger is TIM3_TRGO)
<code>HAL_REMAPTRIGGER_TIM1_ITR3</code>	TIM1 ITR3 trigger remap 0: No remap 1: Remap (TIM1_TRG3 = TIM17_OC)
<code>IS_HAL_REMAPTRIGGER</code>	
<i>SYSCFG registers bit address in the alias region</i>	
<code>SYSCFG_OFFSET</code>	
<code>CFG2_OFFSET</code>	
<code>BYPADDRPAR_BitNumber</code>	
<code>CFG2_BYPADDRPAR_BB</code>	
<i>Fast-mode Plus on GPIO</i>	
<code>SYSCFG_FASTMODEPLUS_PB6</code>	Enable Fast-mode Plus on PB6
<code>SYSCFG_FASTMODEPLUS_PB7</code>	Enable Fast-mode Plus on PB7
<code>SYSCFG_FASTMODEPLUS_PB8</code>	Enable Fast-mode Plus on PB8
<code>SYSCFG_FASTMODEPLUS_PB9</code>	Enable Fast-mode Plus on PB9

6 HAL ADC Generic Driver

6.1 ADC Firmware driver registers structures

6.1.1 __ADC_HandleTypeDefDef

Data Fields

- *ADC_TypeDef * Instance*
- *ADC_InitTypeDef Init*
- *DMA_HandleTypeDef * DMA_Handle*
- *HAL_LockTypeDef Lock*
- *_IO uint32_t State*
- *_IO uint32_t ErrorCode*
- *ADC_InjectionConfigTypeDef InjectionConfig*

Field Documentation

- ***ADC_TypeDef* __ADC_HandleTypeDefDef::Instance***
Register base address
- ***ADC_InitTypeDef __ADC_HandleTypeDefDef::Init***
ADC required parameters
- ***DMA_HandleTypeDef* __ADC_HandleTypeDefDef::DMA_Handle***
Pointer DMA Handler
- ***HAL_LockTypeDef __ADC_HandleTypeDefDef::Lock***
ADC locking object
- ***_IO uint32_t __ADC_HandleTypeDefDef::State***
ADC communication state (bitmap of ADC states)
- ***_IO uint32_t __ADC_HandleTypeDefDef::ErrorCode***
ADC Error code
- ***ADC_InjectionConfigTypeDef __ADC_HandleTypeDefDef::InjectionConfig***
ADC injected channel configuration build-up structure

6.2 ADC Firmware driver API description

6.2.1 ADC peripheral features

- 12-bit, 10-bit, 8-bit or 6-bit configurable resolution (available only on STM32F30xxC devices).
- Interrupt generation at the end of regular conversion, end of injected conversion, and in case of analog watchdog or overrun events.
- Single and continuous conversion modes.
- Scan mode for conversion of several channels sequentially.
- Data alignment with in-built data coherency.
- Programmable sampling time (channel wise)
- ADC conversion of regular group and injected group.
- External trigger (timer or EXTI) with configurable polarity for both regular and injected groups.
- DMA request generation for transfer of conversions data of regular group.
- Multimode dual mode (available on devices with 2 ADCs or more).
- Configurable DMA data storage in Multimode Dual mode (available on devices with 2 DCs or more).

- Configurable delay between conversions in Dual interleaved mode (available on devices with 2 DCs or more).
- ADC calibration
- ADC channels selectable single/differential input (available only on STM32F30xxC devices)
- ADC Injected sequencer&channels configuration context queue (available only on STM32F30xxC devices)
- ADC offset on injected and regular groups (offset on regular group available only on STM32F30xxC devices)
- ADC supply requirements: 2.4 V to 3.6 V at full speed and down to 1.8 V at slower speed.
- ADC input range: from Vref- (connected to Vssa) to Vref+ (connected to Vdda or to an external voltage reference).

6.2.2 How to use this driver

Configuration of top level parameters related to ADC

1. Enable the ADC interface
 - As prerequisite, ADC clock must be configured at RCC top level.
 - For STM32F30x/STM32F33x devices: Two possible clock sources: synchronous clock derived from AHB clock or asynchronous clock derived from ADC dedicated PLL 72MHz. - Synchronous clock is mandatory since used as ADC core clock. Synchronous clock can be used optionally as ADC conversion clock, depending on ADC init structure clock setting. Synchronous clock is configured using macro `_ADCx_CLK_ENABLE()`. - Asynchronous can be used optionally as ADC conversion clock, depending on ADC init structure clock setting. Asynchronous clock is configured using function `HAL_RCCEEx_PeriphCLKConfig()`.
 - For example, in case of device with a single ADC: Into `HAL_ADC_MspInit()` (recommended code location) or with other device clock parameters configuration:


```
_HAL_RCC_ADC1_CLK_ENABLE() (mandatory)
PeriphClkInit.PeriphClockSelection = RCC_PERIPHCLK_ADC (optional, if
ADC conversion from asynchronous clock)
PeriphClkInit.Adc1ClockSelection = RCC_ADC1PLLCLK_DIV1 (optional, if
ADC conversion from asynchronous clock)
HAL_RCCEEx_PeriphCLKConfig(&RCC_PeriphClkInitStructure) (optional, if
ADC conversion from asynchronous clock)
```
 - For example, in case of device with 4 ADCs:


```
if((hadc->Instance == ADC1) || (hadc->Instance == ADC2))
{
    _HAL_RCC_ADC1_CLK_ENABLE() (mandatory)
PeriphClkInit.PeriphClockSelection = RCC_PERIPHCLK_ADC (optional, if
ADC conversion from asynchronous clock)
PeriphClkInit.Adc1ClockSelection = RCC_ADC1PLLCLK_DIV1 (optional,
if ADC conversion from asynchronous clock)
HAL_RCCEEx_PeriphCLKConfig(&RCC_PeriphClkInitStructure) (optional, if
ADC conversion from asynchronous clock)
}
else
{
    _HAL_RCC_ADC34_CLK_ENABLE() (mandatory)
PeriphClkInit.Adc34ClockSelection = RCC_ADC34PLLCLK_DIV1; (optional,
if ADC conversion from asynchronous clock)
```

- ```

 HAL_RCCEx_PeriphCLKConfig(&RCC_PeriphClkInitStructure); (optional, if
 ADC conversion from asynchronous clock)
 }
 - For STM32F37x devices: One clock setting is mandatory: ADC clock (core and
 conversion clock) from APB2 clock.
 - Example: Into HAL_ADC_MspInit() (recommended code location) or with
 other device clock parameters configuration:
 PeriphClkInit.PeriphClockSelection = RCC_PERIPHCLK_ADC
 PeriphClkInit.AdcClockSelection = RCC_ADCPOLLCLK_DIV2
 HAL_RCCEx_PeriphCLKConfig(&PeriphClkInit)
2. ADC pins configuration
 - Enable the clock for the ADC GPIOs using macro
 __HAL_RCC_GPIOx_CLK_ENABLE()
 - Configure these ADC pins in analog mode using function HAL_GPIO_Init()
3. Optionally, in case of usage of ADC with interruptions:
 - Configure the NVIC for ADC using function HAL_NVIC_EnableIRQ(ADCx_IRQn)
 - Insert the ADC interruption handler function HAL_ADC_IRQHandler() into the
 function of corresponding ADC interruption vector ADCx_IRQHandler().
4. Optionally, in case of usage of DMA:
 - Configure the DMA (DMA channel, mode normal or circular, ...) using function
 HAL_DMA_Init().
 - Configure the NVIC for DMA using function
 HAL_NVIC_EnableIRQ(DMAx_Channelx_IRQn)
 - Insert the ADC interruption handler function HAL_ADC_IRQHandler() into the
 function of corresponding DMA interruption vector
 DMAx_Channelx_IRQHandler().

```

### Configuration of ADC, groups regular/injected, channels parameters

- Configure the ADC parameters (resolution, data alignment, ...) and regular group parameters (conversion trigger, sequencer, ...) using function HAL\_ADC\_Init().
- Configure the channels for regular group parameters (channel number, channel rank into sequencer, ..., into regular group) using function HAL\_ADC\_ConfigChannel().
- Optionally, configure the injected group parameters (conversion trigger, sequencer, ..., of injected group) and the channels for injected group parameters (channel number, channel rank into sequencer, ..., into injected group) using function HAL\_ADCEx\_InjectedConfigChannel().
- Optionally, configure the analog watchdog parameters (channels monitored, thresholds, ...) using function HAL\_ADC\_AnalogWDGConfig().
- Optionally, for devices with several ADC instances: configure the multimode parameters using function HAL\_ADCEx\_MultiModeConfigChannel().

### Execution of ADC conversions

- Optionally, perform an automatic ADC calibration to improve the conversion accuracy using function HAL\_ADCEx\_Calibration\_Start().
- ADC driver can be used among three modes: polling, interruption, transfer by DMA.
  - ADC conversion by polling:
    - Activate the ADC peripheral and start conversions using function HAL\_ADC\_Start()
    - Wait for ADC conversion completion using function HAL\_ADC\_PollForConversion() (or for injected group: HAL\_ADCEx\_InjectedPollForConversion() )
    - Retrieve conversion results using function HAL\_ADC\_GetValue() (or for injected group: HAL\_ADCEx\_InjectedGetValue() )

- Stop conversion and disable the ADC peripheral using function `HAL_ADC_Stop()`
- ADC conversion by interruption:
  - Activate the ADC peripheral and start conversions using function `HAL_ADC_Start_IT()`
  - Wait for ADC conversion completion by call of function `HAL_ADC_ConvCpltCallback()` (this function must be implemented in user program) (or for injected group: `HAL_ADCEx_InjectedConvCpltCallback()`)
  - Retrieve conversion results using function `HAL_ADC_GetValue()` (or for injected group: `HAL_ADCEx_InjectedGetValue()`)
  - Stop conversion and disable the ADC peripheral using function `HAL_ADC_Stop_IT()`
- ADC conversion with transfer by DMA:
  - Activate the ADC peripheral and start conversions using function `HAL_ADC_Start_DMA()`
  - Wait for ADC conversion completion by call of function `HAL_ADC_ConvCpltCallback()` or `HAL_ADC_ConvHalfCpltCallback()` (these functions must be implemented in user program)
  - Conversion results are automatically transferred by DMA into destination variable address.
  - Stop conversion and disable the ADC peripheral using function `HAL_ADC_Stop_DMA()`
- For devices with several ADCs: ADC multimode conversion with transfer by DMA:
  - Activate the ADC peripheral (slave) using function `HAL_ADC_Start()` (conversion start pending ADC master)
  - Activate the ADC peripheral (master) and start conversions using function `HAL_ADCEx_MultiModeStart_DMA()`
  - Wait for ADC conversion completion by call of function `HAL_ADC_ConvCpltCallback()` or `HAL_ADC_ConvHalfCpltCallback()` (these functions must be implemented in user program)
  - Conversion results are automatically transferred by DMA into destination variable address.
  - Stop conversion and disable the ADC peripheral (master) using function `HAL_ADCEx_MultiModeStop_DMA()`
  - Stop conversion and disable the ADC peripheral (slave) using function `HAL_ADC_Stop_IT()`



Callback functions must be implemented in user program:

- `HAL_ADC_ErrorCallback()`
- `HAL_ADC_LevelOutOfWindowCallback()` (callback of analog watchdog)
- `HAL_ADC_ConvCpltCallback()`
- `HAL_ADC_ConvHalfCpltCallback()`
- `HAL_ADCEx_InjectedConvCpltCallback()`
- `HAL_ADCEx_InjectedQueueOverflowCallback()` (for STM32F30x/STM32F33x devices)

## Deinitialization of ADC

1. Disable the ADC interface
  - ADC clock can be hard reset and disabled at RCC top level.
  - Hard reset of ADC peripherals using macro `_ADCx_FORCE_RESET()`, `_ADCx_RELEASE_RESET()`.
  - ADC clock disable using the equivalent macro/functions as configuration step.
  - For STM32F30x/STM32F33x devices: Caution: For devices with several ADCs: These settings impact both ADC of common group: ADC1&ADC2, ADC3&ADC4 if available (ADC2, ADC3, ADC4 availability depends on STM32 product)
    - For example, in case of device with a single ADC: Into `HAL_ADC_MspDeInit()` (recommended code location) or with other device clock parameters configuration:
      - `_HAL_RCC_ADC1_FORCE_RESET()` (optional)
      - `_HAL_RCC_ADC1_RELEASE_RESET()` (optional)
      - `_HAL_RCC_ADC1_CLK_DISABLE()` (mandatory)
    - PeriphClkInit.PeriphClockSelection = `RCC_PERIPHCLK_ADC` (optional, if configured before)
    - PeriphClkInit.Adc1ClockSelection = `RCC_ADC1PLLCLK_OFF` (optional, if configured before)
    - `HAL_RCCEEx_PeriphCLKConfig(&RCC_PeriphClkInitStructure)` (optional, if configured before)
  - For example, in case of device with 4 ADCs:
 

```
if((hadc->Instance == ADC1) || (hadc->Instance == ADC2))
{
 _HAL_RCC_ADC12_FORCE_RESET() (optional)
 _HAL_RCC_ADC12_RELEASE_RESET() (optional)
 _HAL_RCC_ADC12_CLK_DISABLE() (mandatory)
}
else
{
 _HAL_RCC_ADC32_FORCE_RESET() (optional)
 _HAL_RCC_ADC32_RELEASE_RESET() (optional)
 _HAL_RCC_ADC34_CLK_DISABLE() (mandatory)
}
```

    - PeriphClkInit.PeriphClockSelection = `RCC_PERIPHCLK_ADC` (optional, if configured before)
    - PeriphClkInit.Adc12ClockSelection = `RCC_ADC12PLLCLK_OFF` (optional, if configured before)
    - `HAL_RCCEEx_PeriphCLKConfig(&RCC_PeriphClkInitStructure)` (optional, if configured before)
  - For STM32F37x devices:
    - Example: Into `HAL_ADC_MspDeInit()` (recommended code location) or with other device clock parameters configuration:
 

```
PeriphClkInit.PeriphClockSelection = RCC_PERIPHCLK_ADC
PeriphClkInit.AdcClockSelection = RCC_ADCPLLCLK_OFF
HAL_RCCEEx_PeriphCLKConfig(&PeriphClkInit)
```
2. ADC pins configuration
  - Disable the clock for the ADC GPIOs using macro `_HAL_RCC_GPIOx_CLK_DISABLE()`

3. Optionally, in case of usage of ADC with interruptions:
  - Disable the NVIC for ADC using function HAL\_NVIC\_EnableIRQ(ADCx\_IRQHandler)
4. Optionally, in case of usage of DMA:
  - Deinitialize the DMA using function HAL\_DMA\_Init().
  - Disable the NVIC for DMA using function HAL\_NVIC\_DisableIRQ(DMAx\_Channelx\_IRQHandler)

### 6.2.3 Initialization and de-initialization functions

This section provides functions allowing to:

- Initialize and configure the ADC.
- De-initialize the ADC.

This section contains the following APIs:

- [\*HAL\\_ADC\\_Init\(\)\*](#)
- [\*HAL\\_ADC\\_DeInit\(\)\*](#)
- [\*HAL\\_ADC\\_MspInit\(\)\*](#)
- [\*HAL\\_ADC\\_MspDeInit\(\)\*](#)

### 6.2.4 IO operation functions

This section provides functions allowing to:

- Start conversion of regular group.
- Stop conversion of regular group.
- Poll for conversion complete on regular group.
- Poll for conversion event.
- Get result of regular channel conversion.
- Start conversion of regular group and enable interruptions.
- Stop conversion of regular group and disable interruptions.
- Handle ADC interrupt request
- Start conversion of regular group and enable DMA transfer.
- Stop conversion of regular group and disable ADC DMA transfer.

This section contains the following APIs:

- [\*HAL\\_ADC\\_Start\(\)\*](#)
- [\*HAL\\_ADC\\_Stop\(\)\*](#)
- [\*HAL\\_ADC\\_PollForConversion\(\)\*](#)
- [\*HAL\\_ADC\\_PollForEvent\(\)\*](#)
- [\*HAL\\_ADC\\_Start\\_IT\(\)\*](#)
- [\*HAL\\_ADC\\_Stop\\_IT\(\)\*](#)
- [\*HAL\\_ADC\\_Start\\_DMA\(\)\*](#)
- [\*HAL\\_ADC\\_Stop\\_DMA\(\)\*](#)
- [\*HAL\\_ADC\\_GetValue\(\)\*](#)
- [\*HAL\\_ADC\\_IRQHandler\(\)\*](#)
- [\*HAL\\_ADC\\_ConvCpltCallback\(\)\*](#)
- [\*HAL\\_ADC\\_ConvHalfCpltCallback\(\)\*](#)
- [\*HAL\\_ADC\\_LevelOutOfWindowCallback\(\)\*](#)
- [\*HAL\\_ADC\\_ErrorCallback\(\)\*](#)

## 6.2.5 Peripheral Control functions

This section provides functions allowing to:

- Configure channels on regular group
- Configure the analog watchdog

This section contains the following APIs:

- [\*HAL\\_ADC\\_ConfigChannel\(\)\*](#)
- [\*HAL\\_ADC\\_AnalogWDGConfig\(\)\*](#)

## 6.2.6 Peripheral state and errors functions

This subsection provides functions to get in run-time the status of the peripheral.

- Check the ADC state
- Check the ADC error code

This section contains the following APIs:

- [\*HAL\\_ADC\\_GetState\(\)\*](#)
- [\*HAL\\_ADC\\_GetError\(\)\*](#)

## 6.2.7 Detailed description of functions

### **HAL\_ADC\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADC_Init (ADC_HandleTypeDef * hadc)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description | Initializes the ADC peripheral and regular group according to parameters specified in structure "ADC_InitTypeDef".                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Notes                | <ul style="list-style-type: none"> <li>• As prerequisite, ADC clock must be configured at RCC top level depending on both possible clock sources: PLL clock or AHB clock. See commented example code below that can be copied and uncommented into HAL_ADC_MspInit().</li> <li>• Possibility to update parameters on the fly: This function initializes the ADC MSP (HAL_ADC_MspInit()) only when coming from ADC state reset. Following calls to this function can be used to reconfigure some parameters of ADC_InitTypeDef structure on the fly, without modifying MSP configuration. If ADC MSP has to be modified again, HAL_ADC_DeInit() must be called before HAL_ADC_Init(). The setting of these parameters is conditioned to ADC state. For parameters constraints, see comments of structure "ADC_InitTypeDef".</li> <li>• This function configures the ADC within 2 scopes: scope of entire ADC and scope of regular group. For parameters details, see comments of structure "ADC_InitTypeDef".</li> <li>• For devices with several ADCs: parameters related to common ADC registers (ADC clock mode) are set only if all ADCs sharing the same common group are disabled. If this is not the case, these common parameters setting are bypassed</li> </ul> |

without error reporting: it can be the intended behaviour in case of update of a parameter of ADC\_InitTypeDef on the fly, without disabling the other ADCs sharing the same common group.

### **HAL\_ADC\_DeInit**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADC_DeInit (ADC_HandleTypeDef * hadc)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Deinitialize the ADC peripheral registers to their default reset values, with deinitialization of the ADC MSP.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Notes                | <ul style="list-style-type: none"> <li>• For devices with several ADCs: reset of ADC common registers is done only if all ADCs sharing the same common group are disabled. If this is not the case, reset of these common parameters reset is bypassed without error reporting: it can be the intended behaviour in case of reset of a single ADC while the other ADCs sharing the same common group is still running.</li> <li>• For devices with several ADCs: Global reset of all ADCs sharing a common group is possible. As this function is intended to reset a single ADC, to not impact other ADCs, instructions for global reset of multiple ADCs have been let commented below. If needed, the example code can be copied and uncommented into function HAL_ADC_MspDeInit().</li> </ul> |

### **HAL\_ADC\_MspInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_ADC_MspInit (ADC_HandleTypeDef * hadc)</b>                      |
| Function description | Initializes the ADC MSP.                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_ADC\_MspDeInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_ADC_MspDeInit (ADC_HandleTypeDef * hadc)</b>                    |
| Function description | Deinitializes the ADC MSP.                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_ADC\_Start**

|                      |                                                                   |
|----------------------|-------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADC_Start (ADC_HandleTypeDef * hadc)</b> |
| Function description | Enables ADC, starts conversion of regular group.                  |

|               |                                                                                                                                                                                                                                                                                                                       |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li><b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                                                                             |
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                  |
| Notes         | <ul style="list-style-type: none"> <li>: Case of multimode enabled (for devices with several ADCs): This function must be called for ADC slave first, then ADC master. For ADC slave, ADC is enabled only (conversion is not started). For ADC master, ADC is enabled and multimode conversion is started.</li> </ul> |

### HAL\_ADC\_Stop

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADC_Stop (ADC_HandleTypeDef * hadc)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Stop ADC conversion of regular group (and injected group in case of auto_injection mode), disable ADC peripheral.                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li><b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Notes                | <ul style="list-style-type: none"> <li>: ADC peripheral disable is forcing stop of potential conversion on injected group. If injected group is under use, it should be preliminarily stopped using HAL_ADCEx_InjectedStop function.</li> <li>: Case of multimode enabled (for devices with several ADCs): This function must be called for ADC master first, then ADC slave. For ADC master, conversion is stopped and ADC is disabled. For ADC slave, ADC is disabled only (conversion stop of ADC master has already stopped conversion of ADC slave).</li> </ul> |

### HAL\_ADC\_PollForConversion

|                      |                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADC_PollForConversion (ADC_HandleTypeDef * hadc, uint32_t Timeout)</b>                                  |
| Function description | Wait for regular group conversion to be completed.                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li><b>hadc:</b> ADC handle</li> <li><b>Timeout:</b> Timeout value in millisecond.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                             |

### HAL\_ADC\_PollForEvent

|                      |                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADC_PollForEvent (ADC_HandleTypeDef * hadc, uint32_t EventType, uint32_t Timeout)</b>                                                                                                                                                                                                                                                                          |
| Function description | Poll for conversion event.                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>hadc:</b> ADC handle</li> <li><b>EventType:</b> the ADC event type. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– ADC_AWD_EVENT: ADC Analog watchdog 1 event (main analog watchdog, present on all STM32 devices)</li> <li>– ADC_AWD2_EVENT: ADC Analog watchdog 2 event</li> </ul> </li> </ul> |

---

|                   |                                                                                                             |
|-------------------|-------------------------------------------------------------------------------------------------------------|
|                   | (additional analog watchdog, present only on STM32F3 devices)                                               |
|                   | – ADC_AWD3_EVENT: ADC Analog watchdog 3 event (additional analog watchdog, present only on STM32F3 devices) |
|                   | – ADC_OVR_EVENT: ADC Overrun event                                                                          |
|                   | – ADC_JQOVF_EVENT: ADC Injected context queue overflow event                                                |
| • <b>Timeout:</b> | Timeout value in millisecond.                                                                               |
| Return values     | • <b>HAL:</b> status                                                                                        |

### HAL\_ADC\_Start\_IT

|                      |                                                                      |
|----------------------|----------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADC_Start_IT (ADC_HandleTypeDef * hadc)</b> |
| Function description | Enables ADC, starts conversion of regular group with interruption.   |

### HAL\_ADC\_Stop\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADC_Stop_IT (ADC_HandleTypeDef * hadc)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Stop ADC conversion of regular group (and injected group in case of auto_injection mode), disable interruption of end-of-conversion, disable ADC peripheral.                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | • <b>hadc:</b> ADC handle                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Return values        | • <b>HAL:</b> status.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Notes                | <ul style="list-style-type: none"> <li>: ADC peripheral disable is forcing stop of potential conversion on injected group. If injected group is under use, it should be preliminarily stopped using HAL_ADCEx_InjectedStop function.</li> <li>: Case of multimode enabled (for devices with several ADCs): This function must be called for ADC master first, then ADC slave. For ADC master, conversion is stopped and ADC is disabled. For ADC slave, ADC is disabled only (conversion stop of ADC master has already stopped conversion of ADC slave).</li> </ul> |

### HAL\_ADC\_Start\_DMA

|                      |                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADC_Start_DMA (ADC_HandleTypeDef * hadc, uint32_t * pData, uint32_t Length)</b> |
| Function description | Enables ADC, starts conversion of regular group and transfers result through DMA.                        |

### HAL\_ADC\_Stop\_DMA

|                      |                                                                      |
|----------------------|----------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADC_Stop_DMA (ADC_HandleTypeDef * hadc)</b> |
| Function description | Stop ADC conversion of regular group (and injected group in case     |

of auto\_injection mode), disable ADC DMA transfer, disable ADC peripheral.

- |               |                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                                                                                                                                                                                   |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status.</li> </ul>                                                                                                                                                                                                                                                                                                                                                       |
| Notes         | <ul style="list-style-type: none"> <li>• : ADC peripheral disable is forcing stop of potential conversion on injected group. If injected group is under use, it should be preliminarily stopped using HAL_ADCEx_InjectedStop function.</li> <li>• : Case of multimode enabled (for devices with several ADCs): This function is for single-ADC mode only. For multimode, use the dedicated MultimodeStop function.</li> </ul> |

### **HAL\_ADC\_GetValue**

- |                      |                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_ADC_GetValue (ADC_HandleTypeDef * hadc)</b>                                                                                                                                                                                                                 |
| Function description | Get ADC regular group conversion result.                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                                 |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Converted:</b> value</li> </ul>                                                                                                                                                                                                 |
| Notes                | <ul style="list-style-type: none"> <li>• Reading DR register automatically clears EOC (end of conversion of regular group) flag. Additionally, this functions clears EOS (end of sequence of regular group) flag, in case of the end of the sequence is reached.</li> </ul> |

### **HAL\_ADC\_IRQHandler**

- |                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_ADC_IRQHandler (ADC_HandleTypeDef * hadc)</b>                   |
| Function description | Handles ADC interrupt request.                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_ADC\_ConvCpltCallback**

- |                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_ADC_ConvCpltCallback (ADC_HandleTypeDef * hadc)</b>             |
| Function description | Conversion complete callback in non blocking mode.                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_ADC\_ConvHalfCpltCallback**

- |                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_ADC_ConvHalfCpltCallback (ADC_HandleTypeDef * hadc)</b>         |
| Function description | Conversion DMA half-transfer callback in non blocking mode.                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_ADC\_LevelOutOfWindowCallback**

Function name      **void HAL\_ADC\_LevelOutOfWindowCallback  
(ADC\_HandleTypeDef \* hadc)**

Function description      Analog watchdog callback in non blocking mode.

Parameters      • **hadc:** ADC handle

Return values      • **None**

**HAL\_ADC\_ErrorCallback**

Function name      **void HAL\_ADC\_ErrorCallback (ADC\_HandleTypeDef \* hadc)**

Function description      ADC error callback in non blocking mode (ADC conversion with interruption or transfer by DMA)

Parameters      • **hadc:** ADC handle

Return values      • **None**

**HAL\_ADC\_ConfigChannel**

Function name      **HAL\_StatusTypeDef HAL\_ADC\_ConfigChannel  
(ADC\_HandleTypeDef \* hadc, ADC\_ChannelConfTypeDef \*  
sConfig)**

Function description      Configures the the selected channel to be linked to the regular group.

Parameters      • **hadc:** ADC handle  
• **sConfig:** Structure of ADC channel for regular group.

Return values      • **HAL:** status

Notes      • In case of usage of internal measurement channels:  
Vbat/VrefInt/TempSensor. The recommended sampling time  
is at least: For devices STM32F37x: 17.1us for temperature  
sensorFor the other STM32F3 devices: 2.2us for each of  
channels Vbat/VrefInt/TempSensor. These internal paths can  
be disabled using function HAL\_ADC\_DelInit().  
• Possibility to update parameters on the fly: This function  
initializes channel into regular group, following calls to this  
function can be used to reconfigure some parameters of  
structure "ADC\_ChannelConfTypeDef" on the fly, without  
reseting the ADC. The setting of these parameters is  
conditioned to ADC state. For parameters constraints, see  
comments of structure "ADC\_ChannelConfTypeDef".

**HAL\_ADC\_AnalogWDGConfig**

Function name      **HAL\_StatusTypeDef HAL\_ADC\_AnalogWDGConfig  
(ADC\_HandleTypeDef \* hadc, ADC\_AnalogWDGConfTypeDef \*  
AnalogWDGConfig)**

Function description      Configures the analog watchdog.

Parameters      • **hadc:** ADC handle  
• **AnalogWDGConfig:** Structure of ADC analog watchdog

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | configuration                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                             |
| Notes         | <ul style="list-style-type: none"> <li>• Possibility to update parameters on the fly: This function initializes the selected analog watchdog, following calls to this function can be used to reconfigure some parameters of structure "ADC_AnalogWDGConfTypeDef" on the fly, without resetting the ADC. The setting of these parameters is conditioned to ADC state. For parameters constraints, see comments of structure "ADC_AnalogWDGConfTypeDef".</li> </ul> |

### HAL\_ADC\_GetState

|                      |                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_ADC_GetState (ADC_HandleTypeDef * hadc)</code>                                                                                                                                                                                                                                                                                                         |
| Function description | return the ADC state                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                                                                                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• ADC state machine is managed by bitfield, state must be compared with bit by bit. For example: " if<br/> <code>(HAL_IS_BIT_SET(HAL_ADC_GetState(hadc1),</code><br/> <code>HAL_ADC_STATE_REG_BUSY)) " " if</code><br/> <code>(HAL_IS_BIT_SET(HAL_ADC_GetState(hadc1),</code><br/> <code>HAL_ADC_STATE_AWD1) ) "</code></li> </ul> |

### HAL\_ADC\_GetError

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_ADC_GetError (ADC_HandleTypeDef * hadc)</code>           |
| Function description | Return the ADC error code.                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>ADC:</b> Error Code</li> </ul>  |

## 6.3 ADC Firmware driver defines

### 6.3.1 ADC

#### *ADC Calibration Factor Length Verification*

|                |                                                                                                                 |
|----------------|-----------------------------------------------------------------------------------------------------------------|
| IS_ADC_CALFACT | <b>Description:</b>                                                                                             |
|                | <ul style="list-style-type: none"> <li>• Calibration factor length verification (7 bits maximum)</li> </ul>     |
|                | <b>Parameters:</b>                                                                                              |
|                | <ul style="list-style-type: none"> <li>• <code>_Calibration_Factor_</code>: Calibration factor value</li> </ul> |
|                | <b>Return value:</b>                                                                                            |
|                | <ul style="list-style-type: none"> <li>• None</li> </ul>                                                        |

#### *ADC Conversion Group*

`ADC_REGULAR_GROUP`  
`ADC_INJECTED_GROUP`

## ADC\_REGULAR\_INJECTED\_GROUP

**ADC Exported Macros**

| <code>_HAL_ADC_RESET_HANDLE_STATE</code> | <b>Description:</b>                                                                   |
|------------------------------------------|---------------------------------------------------------------------------------------|
|                                          | <ul style="list-style-type: none"> <li>• Reset ADC handle state.</li> </ul>           |
|                                          | <b>Parameters:</b>                                                                    |
|                                          | <ul style="list-style-type: none"> <li>• <code>_HANDLE_</code>: ADC handle</li> </ul> |
|                                          | <b>Return value:</b>                                                                  |
|                                          | <ul style="list-style-type: none"> <li>• None</li> </ul>                              |

**ADC Exported Types**

| <code>HAL_ADC_STATE_RESET</code>          | <b>Notes:</b>                                                                                                                                                                                                                                                                                                                            |
|-------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                           | <ul style="list-style-type: none"> <li>• ADC state machine is managed by bitfields, state must be compared with bit by bit. For example: " if (HAL_IS_BIT_SET(HAL_ADC_GetState(had c1), HAL_ADC_STATE_REG_BUSY)) " " if (HAL_IS_BIT_SET(HAL_ADC_GetState(had c1), HAL_ADC_STATE_AWD1) ) " ADC not yet initialized or disabled</li> </ul> |
| <code>HAL_ADC_STATE_READY</code>          | ADC peripheral ready for use                                                                                                                                                                                                                                                                                                             |
| <code>HAL_ADC_STATE_BUSY_INTERNAL</code>  | ADC is busy to internal process (initialization, calibration)                                                                                                                                                                                                                                                                            |
| <code>HAL_ADC_STATE_TIMEOUT</code>        | TimeOut occurrence                                                                                                                                                                                                                                                                                                                       |
| <code>HAL_ADC_STATE_ERROR_INTERNAL</code> | Internal error occurrence                                                                                                                                                                                                                                                                                                                |
| <code>HAL_ADC_STATE_ERROR_CONFIG</code>   | Configuration error occurrence                                                                                                                                                                                                                                                                                                           |
| <code>HAL_ADC_STATE_ERROR_DMA</code>      | DMA error occurrence                                                                                                                                                                                                                                                                                                                     |
| <code>HAL_ADC_STATE_REG_BUSY</code>       | A conversion on group regular is ongoing or can occur (either by continuous mode, external trigger, low power auto power-on, multimode ADC master control)                                                                                                                                                                               |
| <code>HAL_ADC_STATE_REG_EOC</code>        | Conversion data available on group regular                                                                                                                                                                                                                                                                                               |
| <code>HAL_ADC_STATE_REG_OVR</code>        | Overrun occurrence                                                                                                                                                                                                                                                                                                                       |
| <code>HAL_ADC_STATE_REG_EOSMP</code>      | End Of Sampling flag raised                                                                                                                                                                                                                                                                                                              |
| <code>HAL_ADC_STATE_INJ_BUSY</code>       | A conversion on group injected is ongoing or can occur (either by auto-injection mode, external trigger, low power auto power-on, multimode ADC master control)                                                                                                                                                                          |
| <code>HAL_ADC_STATE_INJ_EOC</code>        | Conversion data available on group injected                                                                                                                                                                                                                                                                                              |
| <code>HAL_ADC_STATE_INJ_JQOVF</code>      | Injected queue overflow occurrence                                                                                                                                                                                                                                                                                                       |
| <code>HAL_ADC_STATE_AWD1</code>           | Out-of-window occurrence of analog watchdog 1                                                                                                                                                                                                                                                                                            |

|                               |                                                                  |
|-------------------------------|------------------------------------------------------------------|
| HAL_ADC_STATE_AWD2            | Out-of-window occurrence of analog watchdog 2                    |
| HAL_ADC_STATE_AWD3            | Out-of-window occurrence of analog watchdog 3                    |
| HAL_ADC_STATE_MULTIMODE_SLAVE | ADC in multimode slave state, controlled by another ADC master ( |

***ADC Injected Conversion Number Verification***

IS\_ADC\_INJECTED\_NB\_CONV

***ADC Regular Discontinuous Mode Number Verification***

IS\_ADC\_REGULAR\_DISCONT\_NUMBER

***ADC Regular Conversion Number Verification***

IS\_ADC\_REGULAR\_NB\_CONV

## 7 HAL ADC Extension Driver

### 7.1 ADCEx Firmware driver registers structures

#### 7.1.1 ADC\_InitTypeDef

##### Data Fields

- *uint32\_t ClockPrescaler*
- *uint32\_t Resolution*
- *uint32\_t DataAlign*
- *uint32\_t ScanConvMode*
- *uint32\_t EOCSelection*
- *uint32\_t LowPowerAutoWait*
- *uint32\_t ContinuousConvMode*
- *uint32\_t NbrOfConversion*
- *uint32\_t DiscontinuousConvMode*
- *uint32\_t NbrOfDiscConversion*
- *uint32\_t ExternalTrigConv*
- *uint32\_t ExternalTrigConvEdge*
- *uint32\_t DMAContinuousRequests*
- *uint32\_t Overrun*

##### Field Documentation

- ***uint32\_t ADC\_InitTypeDef::ClockPrescaler***  
Select ADC clock source (synchronous clock derived from AHB clock or asynchronous clock derived from ADC dedicated PLL 72MHz) and clock prescaler. The clock is common for all the ADCs. This parameter can be a value of [\*\*ADCEx\\_ClockPrescaler\*\*](#)  
Note: In case of usage of channels on injected group, ADC frequency should be lower than AHB clock frequency /4 for resolution 12 or 10 bits, AHB clock frequency /3 for resolution 8 bits, AHB clock frequency /2 for resolution 6 bits. Note: In case of usage of the ADC dedicated PLL clock, this clock must be preliminarily enabled and prescaler set at RCC top level. Note: This parameter can be modified only if all ADCs of the common ADC group are disabled (for products with several ADCs)
- ***uint32\_t ADC\_InitTypeDef::Resolution***  
Configures the ADC resolution. This parameter can be a value of [\*\*ADCEx\\_Resolution\*\*](#)
- ***uint32\_t ADC\_InitTypeDef::DataAlign***  
Specifies ADC data alignment to right (for resolution 12 bits: MSB on register bit 11 and LSB on register bit 0U) (default setting) or to left (for resolution 12 bits, if offset disabled: MSB on register bit 15 and LSB on register bit 4U, if offset enabled: MSB on register bit 14 and LSB on register bit 3U). See reference manual for alignments with other resolutions. This parameter can be a value of [\*\*ADCEx\\_Data\\_align\*\*](#)
- ***uint32\_t ADC\_InitTypeDef::ScanConvMode***  
Configures the sequencer of regular and injected groups. This parameter can be associated to parameter 'DiscontinuousConvMode' to have main sequence subdivided in successive parts. If disabled: Conversion is performed in single mode (one channel converted, the one defined in rank 1U). Parameters 'NbrOfConversion' and 'InjectedNbrOfConversion' are discarded (equivalent to set to 1U). If enabled: Conversions are performed in sequence mode (multiple ranks defined by 'NbrOfConversion'/'InjectedNbrOfConversion' and each channel rank). Scan direction is upward: from rank1 to rank 'n'. This parameter can be a value of [\*\*ADCEx\\_Scan\\_mode\*\*](#)

- ***uint32\_t ADC\_InitTypeDef::EOCSelection***  
Specifies what EOC (End Of Conversion) flag is used for conversion by polling and interruption: end of conversion of each rank or complete sequence. This parameter can be a value of [\*\*ADCEx\\_EOCSelection\*\*](#).
- ***uint32\_t ADC\_InitTypeDef::LowPowerAutoWait***  
Selects the dynamic low power Auto Delay: ADC conversions are performed only when necessary. New conversion starts only when the previous conversion (for regular group) or previous sequence (for injected group) has been treated by user software. This feature automatically adapts the speed of ADC to the speed of the system that reads the data. Moreover, this avoids risk of overrun for low frequency applications. This parameter can be set to ENABLE or DISABLE. Note: Do not use with interruption or DMA (**HAL\_ADC\_Start\_IT()**, **HAL\_ADC\_Start\_DMA()**) since they have to clear immediately the EOC flag to free the IRQ vector sequencer. Do use with polling: 1. Start conversion with **HAL\_ADC\_Start()**, 2. Later on, when conversion data is needed: use **HAL\_ADC\_PollForConversion()** to ensure that conversion is completed and use **HAL\_ADC\_GetValue()** to retrieve conversion result and trig another conversion (in case of usage of injected group, use the equivalent functions **HAL\_ADCExInjected\_Start()**, **HAL\_ADCEx\_InjectedGetValue()**, ...).
- ***uint32\_t ADC\_InitTypeDef::ContinuousConvMode***  
Specifies whether the conversion is performed in single mode (one conversion) or continuous mode for regular group, after the selected trigger occurred (software start or external trigger). This parameter can be set to ENABLE or DISABLE.
- ***uint32\_t ADC\_InitTypeDef::NbrOfConversion***  
Specifies the number of ranks that will be converted within the regular group sequencer. To use the regular group sequencer and convert several ranks, parameter 'ScanConvMode' must be enabled. This parameter must be a number between Min\_Data = 1 and Max\_Data = 16. Note: This parameter must be modified when no conversion is on going on regular group (ADC disabled, or ADC enabled without continuous mode or external trigger that could launch a conversion).
- ***uint32\_t ADC\_InitTypeDef::DiscontinuousConvMode***  
Specifies whether the conversions sequence of regular group is performed in Complete-sequence/Discontinuous-sequence (main sequence subdivided in successive parts). Discontinuous mode is used only if sequencer is enabled (parameter 'ScanConvMode'). If sequencer is disabled, this parameter is discarded. Discontinuous mode can be enabled only if continuous mode is disabled. If continuous mode is enabled, this parameter setting is discarded. This parameter can be set to ENABLE or DISABLE.
- ***uint32\_t ADC\_InitTypeDef::NbrOfDiscConversion***  
Specifies the number of discontinuous conversions in which the main sequence of regular group (parameter NbrOfConversion) will be subdivided. If parameter 'DiscontinuousConvMode' is disabled, this parameter is discarded. This parameter must be a number between Min\_Data = 1 and Max\_Data = 8.
- ***uint32\_t ADC\_InitTypeDef::ExternalTrigConv***  
Selects the external event used to trigger the conversion start of regular group. If set to ADC\_SOFTWARE\_START, external triggers are disabled. This parameter can be a value of [\*\*ADCEx\\_External\\_trigger\\_source-Regular\*\*](#) Caution: For devices with several ADCs, external trigger source is common to ADC common group (for example: ADC1&ADC2, ADC3&ADC4, if available)
- ***uint32\_t ADC\_InitTypeDef::ExternalTrigConvEdge***  
Selects the external trigger edge of regular group. If trigger is set to ADC\_SOFTWARE\_START, this parameter is discarded. This parameter can be a value of [\*\*ADCEx\\_External\\_trigger\\_edge-Regular\*\*](#)
- ***uint32\_t ADC\_InitTypeDef::DMAContinuousRequests***  
Specifies whether the DMA requests are performed in one shot mode (DMA transfer stop when number of conversions is reached) or in Continuous mode (DMA transfer

- unlimited, whatever number of conversions). Note: In continuous mode, DMA must be configured in circular mode. Otherwise an overrun will be triggered when DMA buffer maximum pointer is reached. This parameter can be set to ENABLE or DISABLE.  
 Note: This parameter must be modified when no conversion is on going on both regular and injected groups (ADC disabled, or ADC enabled without continuous mode or external trigger that could launch a conversion).
- ***uint32\_t ADC\_InitTypeDef::Overrun***  
 Select the behaviour in case of overrun: data overwritten (default) or preserved. This parameter is for regular group only. This parameter can be a value of ***ADCEx\_Overrun*** Note: Case of overrun set to data preserved and usage with end on conversion interruption (HAL\_Start\_IT()): ADC IRQ handler has to clear end of conversion flags, this induces the release of the preserved data. If needed, this data can be saved into function **HAL\_ADC\_ConvCpltCallback()** (called before end of conversion flags clear). Note: Error reporting in function of conversion mode:Usage with ADC conversion by polling for event or interruption: Error is reported only if overrun is set to data preserved. If overrun is set to data overwritten, user can willingly not read the conversion data each time, this is not considered as an erroneous case.Usage with ADC conversion by DMA: Error is reported whatever overrun setting (DMA is expected to process all data from data register, any data missed would be abnormal).

## 7.1.2 ADC\_ChannelConfTypeDef

### Data Fields

- ***uint32\_t Channel***
- ***uint32\_t Rank***
- ***uint32\_t SamplingTime***
- ***uint32\_t SingleDiff***
- ***uint32\_t OffsetNumber***
- ***uint32\_t Offset***

### Field Documentation

- ***uint32\_t ADC\_ChannelConfTypeDef::Channel***  
 Specifies the channel to configure into ADC regular group. This parameter can be a value of ***ADCEx\_channels*** Note: Depending on devices, some channels may not be available on package pins. Refer to device datasheet for channels availability.
- ***uint32\_t ADC\_ChannelConfTypeDef::Rank***  
 Specifies the rank in the regular group sequencer. This parameter can be a value of ***ADCEx\_regular\_rank*** Note: In case of need to disable a channel or change order of conversion sequencer, rank containing a previous channel setting can be overwritten by the new channel setting (or parameter number of conversions can be adjusted)
- ***uint32\_t ADC\_ChannelConfTypeDef::SamplingTime***  
 Sampling time value to be set for the selected channel. Unit: ADC clock cycles  
 Conversion time is the addition of sampling time and processing time (12.5 ADC clock cycles at ADC resolution 12 bits, 10.5 cycles at 10 bits, 8.5 cycles at 8 bits, 6.5 cycles at 6 bits). This parameter can be a value of ***ADCEx\_sampling\_times*** Caution: This parameter updates the parameter property of the channel, that can be used into regular and/or injected groups. If this same channel has been previously configured in the other group (regular/injected), it will be updated to last setting. Note: In case of usage of internal measurement channels (VrefInt/Vbat/TempSensor), sampling time constraints must be respected (sampling time can be adjusted in function of ADC clock frequency and sampling time setting) Refer to device datasheet for timings values, parameters TS\_vrefint, TS\_vbat, TS\_temp (values rough order: 2.2us min).
- ***uint32\_t ADC\_ChannelConfTypeDef::SingleDiff***  
 Selection of single-ended or differential input. In differential mode: Differential

measurement is between the selected channel 'i' (positive input) and channel 'i+1' (negative input). Only channel 'i' has to be configured, channel 'i+1' is configured automatically. This parameter must be a value of ***ADCEx\_SingleDifferential*** Caution: This parameter updates the parameter property of the channel, that can be used into regular and/or injected groups. If this same channel has been previously configured in the other group (regular/injected), it will be updated to last setting. Note: Channels 1 to 14 are available in differential mode. Channels 15U, 16U, 17U, 18 can be used only in single-ended mode. Note: When configuring a channel 'i' in differential mode, the channel 'i+1' is not usable separately. Note: This parameter must be modified when ADC is disabled (before ADC start conversion or after ADC stop conversion). If ADC is enabled, this parameter setting is bypassed without error reporting (as it can be the expected behaviour in case of another parameter update on the fly)

- ***uint32\_t ADC\_ChannelConfTypeDef::OffsetNumber***  
Selects the offset number This parameter can be a value of ***ADCEx\_OffsetNumber*** Caution: Only one channel is allowed per channel. If another channel was on this offset number, the offset will be changed to the new channel
- ***uint32\_t ADC\_ChannelConfTypeDef::Offset***  
Defines the offset to be subtracted from the raw converted data when convert channels. Offset value must be a positive number. Depending of ADC resolution selected (12U, 10U, 8 or 6 bits), this parameter must be a number between Min\_Data = 0x000 and Max\_Data = 0xFFFFU, 0x3FFU, 0xFF or 0x3F respectively. Note: This parameter must be modified when no conversion is on going on both regular and injected groups (ADC disabled, or ADC enabled without continuous mode or external trigger that could launch a conversion).

### 7.1.3 ADC\_InjectionConfTypeDef

#### Data Fields

- ***uint32\_t InjectedChannel***
- ***uint32\_t InjectedRank***
- ***uint32\_t InjectedSamplingTime***
- ***uint32\_t InjectedSingleDiff***
- ***uint32\_t InjectedOffsetNumber***
- ***uint32\_t InjectedOffset***
- ***uint32\_t InjectedNbrOfConversion***
- ***uint32\_t InjectedDiscontinuousConvMode***
- ***uint32\_t AutoInjectedConv***
- ***uint32\_t QueueInjectedContext***
- ***uint32\_t ExternalTrigInjecConv***
- ***uint32\_t ExternalTrigInjecConvEdge***

#### Field Documentation

- ***uint32\_t ADC\_InjectionConfTypeDef::InjectedChannel***  
Configure the ADC injected channel This parameter can be a value of ***ADCEx\_channels*** Note: Depending on devices, some channels may not be available on package pins. Refer to device datasheet for channels availability.
- ***uint32\_t ADC\_InjectionConfTypeDef::InjectedRank***  
The rank in the regular group sequencer This parameter must be a value of ***ADCEx\_injected\_rank*** Note: In case of need to disable a channel or change order of conversion sequencer, rank containing a previous channel setting can be overwritten by the new channel setting (or parameter number of conversions can be adjusted)
- ***uint32\_t ADC\_InjectionConfTypeDef::InjectedSamplingTime***  
Sampling time value to be set for the selected channel. Unit: ADC clock cycles  
Conversion time is the addition of sampling time and processing time (12.5 ADC clock

- cycles at ADC resolution 12 bits, 10.5 cycles at 10 bits, 8.5 cycles at 8 bits, 6.5 cycles at 6 bits). This parameter can be a value of ***ADCEx\_sampling\_times*** Caution: This parameter updates the parameter property of the channel, that can be used into regular and/or injected groups. If this same channel has been previously configured in the other group (regular/injected), it will be updated to last setting. Note: In case of usage of internal measurement channels (VrefInt/Vbat/TempSensor), sampling time constraints must be respected (sampling time can be adjusted in function of ADC clock frequency and sampling time setting) Refer to device datasheet for timings values, parameters TS\_vrefint, TS\_vbat, TS\_temp (values rough order: 2.2us min).
- ***uint32\_t ADC\_InjectionConfTypeDef::InjectedSingleDiff***  
Selection of single-ended or differential input. In differential mode: Differential measurement is between the selected channel 'i' (positive input) and channel 'i+1' (negative input). Only channel 'i' has to be configured, channel 'i+1' is configured automatically. This parameter must be a value of ***ADCEx\_SingleDifferential*** Caution: This parameter updates the parameter property of the channel, that can be used into regular and/or injected groups. If this same channel has been previously configured in the other group (regular/injected), it will be updated to last setting. Note: Channels 1 to 14 are available in differential mode. Channels 15U, 16U, 17U, 18 can be used only in single-ended mode. Note: When configuring a channel 'i' in differential mode, the channel 'i-1' is not usable separately. Note: This parameter must be modified when ADC is disabled (before ADC start conversion or after ADC stop conversion). If ADC is enabled, this parameter setting is bypassed without error reporting (as it can be the expected behaviour in case of another parameter update on the fly)
  - ***uint32\_t ADC\_InjectionConfTypeDef::InjectedOffsetNumber***  
Selects the offset number This parameter can be a value of ***ADCEx\_OffsetNumber*** Caution: Only one channel is allowed per offset number. If another channel was on this offset number, the offset will be changed to the new channel.
  - ***uint32\_t ADC\_InjectionConfTypeDef::InjectedOffset***  
Defines the offset to be subtracted from the raw converted data. Offset value must be a positive number. Depending of ADC resolution selected (12U, 10U, 8 or 6 bits), this parameter must be a number between Min\_Data = 0x000 and Max\_Data = 0xFFFFU, 0x3FFU, 0xFF or 0x3F respectively.
  - ***uint32\_t ADC\_InjectionConfTypeDef::InjectedNbrOfConversion***  
Specifies the number of ranks that will be converted within the injected group sequencer. To use the injected group sequencer and convert several ranks, parameter 'ScanConvMode' must be enabled. This parameter must be a number between Min\_Data = 1 and Max\_Data = 4. Caution: this setting impacts the entire injected group. Therefore, call of **HAL\_ADCEx\_InjectedConfigChannel()** to configure a channel on injected group can impact the configuration of other channels previously set.
  - ***uint32\_t ADC\_InjectionConfTypeDef::InjectedDiscontinuousConvMode***  
Specifies whether the conversions sequence of injected group is performed in Complete-sequence/Discontinuous-sequence (main sequence subdivided in successive parts). Discontinuous mode is used only if sequencer is enabled (parameter 'ScanConvMode'). If sequencer is disabled, this parameter is discarded. Discontinuous mode can be enabled only if continuous mode is disabled. If continuous mode is enabled, this parameter setting is discarded. This parameter can be set to ENABLE or DISABLE. Note: This parameter must be modified when ADC is disabled (before ADC start conversion or after ADC stop conversion). Note: For injected group, number of discontinuous ranks increment is fixed to one-by-one. Caution: this setting impacts the entire injected group. Therefore, call of **HAL\_ADCEx\_InjectedConfigChannel()** to configure a channel on injected group can impact the configuration of other channels previously set.
  - ***uint32\_t ADC\_InjectionConfTypeDef::AutoInjectedConv***  
Enables or disables the selected ADC automatic injected group conversion after

regular one This parameter can be set to ENABLE or DISABLE. Note: To use Automatic injected conversion, discontinuous mode must be disabled ('DiscontinuousConvMode' and 'InjectedDiscontinuousConvMode' set to DISABLE) Note: To use Automatic injected conversion, injected group external triggers must be disabled ('ExternalTrigInjecConv' set to ADC\_SOFTWARE\_START) Note: In case of DMA used with regular group: if DMA configured in normal mode (single shot) JAUTO will be stopped upon DMA transfer complete. To maintain JAUTO always enabled, DMA must be configured in circular mode. Caution: this setting impacts the entire injected group. Therefore, call of **HAL\_ADCEx\_InjectedConfigChannel()** to configure a channel on injected group can impact the configuration of other channels previously set.

- ***uint32\_t ADC\_InjectionTypeDef::QueueInjectedContext***

Specifies whether the context queue feature is enabled. This parameter can be set to ENABLE or DISABLE. If context queue is enabled, injected sequencer&channels configurations are queued on up to 2 contexts. If a new injected context is set when queue is full, error is triggered by interruption and through function 'HAL\_ADCEx\_InjectedQueueOverflowCallback'. Caution: This feature request that the sequence is fully configured before injected conversion start. Therefore, configure channels with **HAL\_ADCEx\_InjectedConfigChannel()** as many times as value of 'InjectedNbrOfConversion' parameter. Caution: this setting impacts the entire injected group. Therefore, call of **HAL\_ADCEx\_InjectedConfigChannel()** to configure a channel on injected group can impact the configuration of other channels previously set. Note: This parameter must be modified when ADC is disabled (before ADC start conversion or after ADC stop conversion).

- ***uint32\_t ADC\_InjectionTypeDef::ExternalTrigInjecConv***

Selects the external event used to trigger the conversion start of injected group. If set to ADC\_INJECTED\_SOFTWARE\_START, external triggers are disabled. This parameter can be a value of **ADCEx\_External\_trigger\_source\_Injected** Caution: this setting impacts the entire injected group. Therefore, call of **HAL\_ADCEx\_InjectedConfigChannel()** to configure a channel on injected group can impact the configuration of other channels previously set.

- ***uint32\_t ADC\_InjectionTypeDef::ExternalTrigInjecConvEdge***

Selects the external trigger edge of injected group. This parameter can be a value of **ADCEx\_External\_trigger\_edge\_Injected**. If trigger is set to ADC\_INJECTED\_SOFTWARE\_START, this parameter is discarded. Caution: this setting impacts the entire injected group. Therefore, call of **HAL\_ADCEx\_InjectedConfigChannel()** to configure a channel on injected group can impact the configuration of other channels previously set.

## 7.1.4 ADC\_InjectionConfigTypeDef

### Data Fields

- ***uint32\_t ContextQueue***
- ***uint32\_t ChannelCount***

### Field Documentation

- ***uint32\_t ADC\_InjectionConfigTypeDef::ContextQueue***

Injected channel configuration context: build-up over each **HAL\_ADCEx\_InjectedConfigChannel()** call to finally initialize JSQR register at **HAL\_ADCEx\_InjectedConfigChannel()** last call

- ***uint32\_t ADC\_InjectionConfigTypeDef::ChannelCount***

Number of channels in the injected sequence

## 7.1.5 ADC\_AnalogWDGConfTypeDef

### Data Fields

- *uint32\_t WatchdogNumber*
- *uint32\_t WatchdogMode*
- *uint32\_t Channel*
- *uint32\_t ITMode*
- *uint32\_t HighThreshold*
- *uint32\_t LowThreshold*

### Field Documentation

- ***uint32\_t ADC\_AnalogWDGConfTypeDef::WatchdogNumber***  
Selects which ADC analog watchdog to apply to the selected channel. For Analog Watchdog 1: Only 1 channel can be monitored (or overall group of channels by setting parameter 'WatchdogMode') For Analog Watchdog 2 and 3: Several channels can be monitored (by successive calls of 'HAL\_ADC\_AnalogWDGConfig()' for each channel) This parameter can be a value of [\*ADCEx\\_analog\\_watchdog\\_number\*](#).
- ***uint32\_t ADC\_AnalogWDGConfTypeDef::WatchdogMode***  
For Analog Watchdog 1: Configures the ADC analog watchdog mode: single channel/overall group of channels, regular/injected group. For Analog Watchdog 2 and 3: There is no configuration for overall group of channels as AWD1. Set value 'ADC\_ANALOGWATCHDOG\_NONE' to reset channels group programmed with parameter 'Channel', set any other value to not use this parameter. This parameter can be a value of [\*ADCEx\\_analog\\_watchdog\\_mode\*](#).
- ***uint32\_t ADC\_AnalogWDGConfTypeDef::Channel***  
Selects which ADC channel to monitor by analog watchdog. For Analog Watchdog 1: this parameter has an effect only if parameter 'WatchdogMode' is configured on single channel. Only 1 channel can be monitored. For Analog Watchdog 2 and 3: Several channels can be monitored (successive calls of **HAL\_ADC\_AnalogWDGConfig()** must be done, one for each channel). Channels group reset can be done by setting WatchdogMode to 'ADC\_ANALOGWATCHDOG\_NONE'). This parameter can be a value of [\*ADCEx\\_channels\*](#).
- ***uint32\_t ADC\_AnalogWDGConfTypeDef::ITMode***  
Specifies whether the analog watchdog is configured in interrupt or polling mode. This parameter can be set to ENABLE or DISABLE
- ***uint32\_t ADC\_AnalogWDGConfTypeDef::HighThreshold***  
Configures the ADC analog watchdog High threshold value. Depending of ADC resolution selected (12U, 10U, 8 or 6 bits), this parameter must be a number between Min\_Data = 0x000 and Max\_Data = 0xFFFFU, 0x3FFU, 0xFF or 0x3F respectively.  
Note: Analog watchdog 2 and 3 are limited to a resolution of 8 bits: if ADC resolution is 12 bits the 4 LSB are ignored, if ADC resolution is 10 bits the 2 LSB are ignored.
- ***uint32\_t ADC\_AnalogWDGConfTypeDef::LowThreshold***  
Configures the ADC analog watchdog Low threshold value. Depending of ADC resolution selected (12U, 10U, 8 or 6 bits), this parameter must be a number between Min\_Data = 0x000 and Max\_Data = 0xFFFFU, 0x3FFU, 0xFF or 0x3F respectively.  
Note: Analog watchdog 2 and 3 are limited to a resolution of 8 bits: if ADC resolution is 12 bits the 4 LSB are ignored, if ADC resolution is 10 bits the 2 LSB are ignored.

## 7.1.6 ADC\_MultiModeTypeDef

### Data Fields

- *uint32\_t Mode*
- *uint32\_t DMAAccessMode*
- *uint32\_t TwoSamplingDelay*

### Field Documentation

- ***uint32\_t ADC\_MultiModeTypeDef::Mode***  
Configures the ADC to operate in independent or multi mode. This parameter can be a value of [\*\*ADCEx\\_Common\\_mode\*\*](#)
- ***uint32\_t ADC\_MultiModeTypeDef::DMAAccessMode***  
Configures the DMA mode for multi ADC mode: selection whether 2 DMA channels (each ADC use its own DMA channel) or 1 DMA channel (one DMA channel for both ADC, DMA of ADC master) This parameter can be a value of [\*\*ADCEx\\_Direct\\_memory\\_access\\_mode\\_for\\_multimode\*\*](#) Caution: Limitations with multimode DMA access enabled (1 DMA channel used): In case of dual mode in high speed (more than 5Msps) or high activity of DMA by other peripherals, there is a risk of DMA overrun. Therefore, it is recommended to disable multimode DMA access: each ADC uses its own DMA channel. Refer to device errata sheet for more details.
- ***uint32\_t ADC\_MultiModeTypeDef::TwoSamplingDelay***  
Configures the Delay between 2 sampling phases. This parameter can be a value of [\*\*ADCEx\\_delay\\_between\\_2\\_sampling\\_phases\*\*](#) Delay range depends on selected resolution: from 1 to 12 clock cycles for 12 bits, from 1 to 10 clock cycles for 10 bits from 1 to 8 clock cycles for 8 bits, from 1 to 6 clock cycles for 6 bits

## 7.2 ADCEx Firmware driver API description

### 7.2.1 Initialization and de-initialization functions

This section provides functions allowing to:

- Initialize and configure the ADC.
- De-initialize the ADC.

This section contains the following APIs:

- [\*\*HAL\\_ADC\\_Init\(\)\*\*](#)
- [\*\*HAL\\_ADC\\_DeInit\(\)\*\*](#)

### 7.2.2 IO operation functions

This section provides functions allowing to:

- Start conversion of regular group.
- Stop conversion of regular group.
- Poll for conversion complete on regular group.
- Poll for conversion event.
- Get result of regular channel conversion.
- Start conversion of regular group and enable interruptions.
- Stop conversion of regular group and disable interruptions.
- Handle ADC interrupt request
- Start conversion of regular group and enable DMA transfer.
- Stop conversion of regular group and disable ADC DMA transfer.
- Start conversion of injected group.
- Stop conversion of injected group.
- Poll for conversion complete on injected group.
- Get result of injected channel conversion.
- Start conversion of injected group and enable interruptions.
- Stop conversion of injected group and disable interruptions.
- Start multimode and enable DMA transfer.
- Stop multimode and disable ADC DMA transfer.
- Get result of multimode conversion.

- Perform the ADC self-calibration for single or differential ending.
- Get calibration factors for single or differential ending.
- Set calibration factors for single or differential ending.

This section contains the following APIs:

- *HAL\_ADC\_Start()*
- *HAL\_ADC\_Stop()*
- *HAL\_ADC\_PollForConversion()*
- *HAL\_ADC\_PollForEvent()*
- *HAL\_ADC\_Start\_IT()*
- *HAL\_ADC\_Stop\_IT()*
- *HAL\_ADC\_Start\_DMA()*
- *HAL\_ADC\_Stop\_DMA()*
- *HAL\_ADC\_GetValue()*
- *HAL\_ADC\_IRQHandler()*
- *HAL\_ADCEx\_Calibration\_Start()*
- *HAL\_ADCEx\_Calibration\_GetValue()*
- *HAL\_ADCEx\_Calibration\_SetValue()*
- *HAL\_ADCEx\_InjectedStart()*
- *HAL\_ADCEx\_InjectedStop()*
- *HAL\_ADCEx\_InjectedPollForConversion()*
- *HAL\_ADCEx\_InjectedStart\_IT()*
- *HAL\_ADCEx\_InjectedStop\_IT()*
- *HAL\_ADCEx\_MultiModeStart\_DMA()*
- *HAL\_ADCEx\_MultiModeStop\_DMA()*
- *HAL\_ADCEx\_MultiModeGetValue()*
- *HAL\_ADCEx\_InjectedGetValue()*
- *HAL\_ADCEx-RegularStop()*
- *HAL\_ADCEx-RegularStop\_IT()*
- *HAL\_ADCEx-RegularStop\_DMA()*
- *HAL\_ADCEx-RegularMultiModeStop\_DMA()*
- *HAL\_ADCEx\_InjectedConvCpltCallback()*
- *HAL\_ADCEx\_InjectedQueueOverflowCallback()*
- *HAL\_ADCEx\_LevelOutOfWindow2Callback()*
- *HAL\_ADCEx\_LevelOutOfWindow3Callback()*

### 7.2.3 Peripheral Control functions

This section provides functions allowing to:

- Configure channels on regular group
- Configure channels on injected group
- Configure multimode
- Configure the analog watchdog

This section contains the following APIs:

- *HAL\_ADC\_ConfigChannel()*
- *HAL\_ADCEx\_InjectedConfigChannel()*
- *HAL\_ADC\_AnalogWDGConfig()*
- *HAL\_ADCEx\_MultiModeConfigChannel()*

## 7.2.4 Detailed description of functions

### HAL\_ADCEx\_Calibration\_Start

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_ADCEx_Calibration_Start<br/>(ADC_HandleTypeDef * hadc, uint32_t SingleDiff)</code>                                                                                                                                                                                                                                                                                                               |
| Function description | Perform an ADC automatic self-calibration Calibration prerequisite:<br>ADC must be disabled (execute this function before<br><code>HAL_ADC_Start()</code> or after <code>HAL_ADC_Stop()</code> ).                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> <li>• <b>SingleDiff:</b> Selection of single-ended or differential input This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>ADC_SINGLE_ENDED</code>: Channel in mode input single ended</li> <li>– <code>ADC_DIFFERENTIAL_ENDED</code>: Channel in mode input differential ended</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                       |

### HAL\_ADCEx\_Calibration\_GetValue

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_ADCEx_Calibration_GetValue<br/>(ADC_HandleTypeDef * hadc, uint32_t SingleDiff)</code>                                                                                                                                                                                                                                                                                                                     |
| Function description | Get the calibration factor from automatic conversion result.                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> <li>• <b>SingleDiff:</b> Selection of single-ended or differential input This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>ADC_SINGLE_ENDED</code>: Channel in mode input single ended</li> <li>– <code>ADC_DIFFERENTIAL_ENDED</code>: Channel in mode input differential ended</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Converted:</b> value</li> </ul>                                                                                                                                                                                                                                                                                                                                                  |

### HAL\_ADCEx\_Calibration\_SetValue

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_ADCEx_Calibration_SetValue<br/>(ADC_HandleTypeDef * hadc, uint32_t SingleDiff, uint32_t CalibrationFactor)</code>                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Set the calibration factor to overwrite automatic conversion result.                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> <li>• <b>SingleDiff:</b> Selection of single-ended or differential input This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>ADC_SINGLE_ENDED</code>: Channel in mode input single ended</li> <li>– <code>ADC_DIFFERENTIAL_ENDED</code>: Channel in mode input differential ended</li> </ul> </li> <li>• <b>CalibrationFactor:</b> Calibration factor (coded on 7 bits maximum)</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                          |

**HAL\_ADCEx\_InjectedStart**

|                      |                                                                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx_InjectedStart<br/>(ADC_HandleTypeDef * hadc)</b>                                                                                                                                                                                                                                       |
| Function description | Enables ADC, starts conversion of injected group.                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                                                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                |
| Notes                | <ul style="list-style-type: none"> <li>• Case of multimode enabled (for devices with several ADCs): This function must be called for ADC slave first, then ADC master. For ADC slave, ADC is enabled only (conversion is not started). For ADC master, ADC is enabled and multimode conversion is started.</li> </ul> |

**HAL\_ADCEx\_InjectedStop**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx_InjectedStop<br/>(ADC_HandleTypeDef * hadc)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Stop ADC group injected conversion (potential conversion on going on ADC group regular is not impacted), disable ADC peripheral if no conversion is on going on group regular.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• To stop ADC conversion of both groups regular and injected and to disable ADC peripheral, instead of using 2 functions HAL_ADCEx-RegularStop() and HAL_ADCEx_InjectedStop(), use function HAL_ADC_Stop().</li> <li>• If injected group mode auto-injection is enabled, function HAL_ADC_Stop must be used.</li> <li>• Case of multimode enabled (for devices with several ADCs): This function must be called for ADC master first, then ADC slave. For ADC master, conversion is stopped and ADC is disabled. For ADC slave, ADC is disabled only (conversion stop of ADC master has already stopped conversion of ADC slave).</li> <li>• In case of auto-injection mode, HAL_ADC_Stop must be used.</li> </ul> |

**HAL\_ADCEx\_InjectedPollForConversion**

|                      |                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx_InjectedPollForConversion<br/>(ADC_HandleTypeDef * hadc, uint32_t Timeout)</b>                        |
| Function description | Wait for injected group conversion to be completed.                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> <li>• <b>Timeout:</b> Timeout value in millisecond.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                               |

**HAL\_ADCEx\_InjectedStart\_IT**

|                      |                                                                                    |
|----------------------|------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx_InjectedStart_IT<br/>(ADC_HandleTypeDef * hadc)</b> |
| Function description | Enables ADC, starts conversion of injected group with interruption.                |

**HAL\_ADCEx\_InjectedStop\_IT**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx_InjectedStop_IT<br/>(ADC_HandleTypeDef * hadc)</b>                                                                                              |
| Function description | Stop ADC group injected conversion (potential conversion on going on ADC group regular is not impacted), disable ADC peripheral if no conversion is on going on group regular. |

**HAL\_ADCEx\_MultiModeStart\_DMA**

|                      |                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx_MultiModeStart_DMA<br/>(ADC_HandleTypeDef * hadc, uint32_t * pData, uint32_t Length)</b>             |
| Function description | With ADC configured in multimode, for ADC master: Enables ADC, starts conversion of regular group and transfers result through DMA. |

**HAL\_ADCEx\_MultiModeStop\_DMA**

|                      |                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx_MultiModeStop_DMA<br/>(ADC_HandleTypeDef * hadc)</b>                                                                                                                                                                         |
| Function description | With ADC configured in multimode, for ADC master: Stop ADC group regular conversion (potential conversion on going on ADC group injected is not impacted), disable ADC DMA transfer, disable ADC peripheral if no conversion is on going on group injected. |

**HAL\_ADCEx\_MultiModeGetValue**

|                      |                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_ADCEx_MultiModeGetValue<br/>(ADC_HandleTypeDef * hadc)</b>                                                                                                                                     |
| Function description | Returns the last ADC Master&Slave regular conversions results data in the selected multi mode.                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle of ADC master (handle of ADC slave must not be used)</li> </ul>                                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>The:</b> converted data value.</li> </ul>                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• Reading register CDR does not clear flag ADC flag EOC (ADC group regular end of unitary conversion), as it is the case for independent mode data register.</li> </ul> |

**HAL\_ADCEx-RegularStop**

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx-RegularStop<br/>(ADC_HandleTypeDef * hadc)</b> |
| Function description | Stop ADC group regular conversion (potential conversion on going              |

on ADC group injected is not impacted), disable ADC peripheral if no conversion is on going on group injected.

- |               |                                                                                                                                                                                                                                                                                                                                                                    |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                                                                                                                        |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status.</li> </ul>                                                                                                                                                                                                                                                                                            |
| Notes         | <ul style="list-style-type: none"> <li>• To stop ADC conversion of both groups regular and injected and to disable ADC peripheral, instead of using 2 functions HAL_ADCEx_RegularStop() and HAL_ADCEx_InjectedStop(), use function HAL_ADC_Stop().</li> <li>• In case of auto-injection mode, this function also stop conversion on ADC group injected.</li> </ul> |

### **HAL\_ADCEx-RegularStop\_IT**

|                      |                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx-RegularStop_IT<br/>(ADC_HandleTypeDef * hadc)</b>                                                                                                |
| Function description | Stop ADC group regular conversion (potential conversion on going on ADC group injected is not impacted), disable ADC peripheral if no conversion is on going on group injected. |

### **HAL\_ADCEx-RegularStop\_DMA**

|                      |                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx-RegularStop_DMA<br/>(ADC_HandleTypeDef * hadc)</b>                                                                                                                         |
| Function description | Stop ADC group regular conversion (potential conversion on going on ADC group injected is not impacted), disable ADC DMA transfer, disable ADC peripheral if no conversion is on going on group injected. |

### **HAL\_ADCEx-RegularMultiModeStop\_DMA**

|                      |                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef<br/>HAL_ADCEx-RegularMultiModeStop_DMA<br/>(ADC_HandleTypeDef * hadc)</b>                                                                                                                                                              |
| Function description | With ADC configured in multimode, for ADC master: Stop ADC group regular conversion (potential conversion on going on ADC group injected is not impacted), disable ADC DMA transfer, disable ADC peripheral if no conversion is on going on group injected. |

### **HAL\_ADCEx-InjectedGetValue**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_ADCEx-InjectedGetValue (ADC_HandleTypeDef<br/>* hadc, uint32_t InjectedRank)</b>                                                                                                                                                                                                                                                                                                                           |
| Function description | Get ADC injected group conversion result.                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> <li>• <b>InjectedRank:</b> the converted ADC injected rank. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– ADC_INJECTED_RANK_1: Injected Channel1 selected</li> <li>– ADC_INJECTED_RANK_2: Injected Channel2 selected</li> <li>– ADC_INJECTED_RANK_3: Injected Channel3 selected</li> </ul> </li> </ul> |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>- ADC_INJECTED_RANK_4: Injected Channel4 selected</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Return values | <ul style="list-style-type: none"> <li>• <b>ADC:</b> group injected conversion data</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes         | <ul style="list-style-type: none"> <li>• Reading register JDRx automatically clears ADC flag JEOC (ADC group injected end of unitary conversion).</li> <li>• This function does not clear ADC flag JEOS (ADC group injected end of sequence conversion) Occurrence of flag JEOS rising: If sequencer is composed of 1 rank, flag JEOS is equivalent to flag JEOC. If sequencer is composed of several ranks, during the scan sequence flag JEOC only is raised, at the end of the scan sequence both flags JEOC and EOS are raised. Flag JEOS must not be cleared by this function because it would not be compliant with low power features (feature low power auto-wait, not available on all STM32 families). To clear this flag, either use function: in programming model IT: HAL_ADC_IRQHandler(), in programming model polling:<br/>HAL_ADCEx_InjectedPollForConversion() or<br/>__HAL_ADC_CLEAR_FLAG(&amp;hadc, ADC_FLAG_JEOS).</li> </ul> |

### **HAL\_ADCEx\_InjectedConvCpltCallback**

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <b>void HAL_ADCEx_InjectedConvCpltCallback<br/>(ADC_HandleTypeDef * hadc)</b> |
| Function description | Injected conversion complete callback in non blocking mode.                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>               |

### **HAL\_ADCEx\_InjectedQueueOverflowCallback**

|                      |                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_ADCEx_InjectedQueueOverflowCallback<br/>(ADC_HandleTypeDef * hadc)</b>                                                                                                                                                                                  |
| Function description | Injected context queue overflow flag callback.                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>                                                                                                                                                                                         |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This callback is called if injected context queue is enabled (parameter "QueueInjectedContext" in injected channel configuration) and if a new injected context is set when queue is full (maximum 2 contexts).</li> </ul> |

### **HAL\_ADCEx\_LevelOutOfWindow2Callback**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>void HAL_ADCEx_LevelOutOfWindow2Callback<br/>(ADC_HandleTypeDef * hadc)</b> |
| Function description | Analog watchdog 2 callback in non blocking mode.                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> </ul>    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                |

**HAL\_ADCEx\_LevelOutOfWindow3Callback**

Function name      **void HAL\_ADCEx\_LevelOutOfWindow3Callback  
(ADC\_HandleTypeDef \* hadc)**

Function description      Analog watchdog 3 callback in non blocking mode.

Parameters      • **hadc:** ADC handle

Return values      • **None**

**HAL\_ADCEx\_InjectedConfigChannel**

Function name      **HAL\_StatusTypeDef HAL\_ADCEx\_InjectedConfigChannel  
(ADC\_HandleTypeDef \* hadc, ADC\_InjectionConfTypeDef \*  
sConfigInjected)**

Function description      Configures the ADC injected group and the selected channel to be linked to the injected group.

Parameters      • **hadc:** ADC handle  
• **sConfigInjected:** Structure of ADC injected group and ADC channel for injected group.

Return values      • **None**

Notes      • Possibility to update parameters on the fly: This function initializes injected group, following calls to this function can be used to reconfigure some parameters of structure "ADC\_InjectionConfTypeDef" on the fly, without resetting the ADC. The setting of these parameters is conditioned to ADC state. For parameters constraints, see comments of structure "ADC\_InjectionConfTypeDef".  
 • In case of usage of internal measurement channels: Vbat/VrefInt/TempSensor. The recommended sampling time is at least: For devices STM32F37x: 17.1us for temperature sensorFor the other STM32F3 devices: 2.2us for each of channels Vbat/VrefInt/TempSensor. These internal paths can be disabled using function HAL\_ADC\_DeInit().  
 • To reset injected sequencer, function HAL\_ADCEx\_InjectedStop() can be used.  
 • Caution: For Injected Context Queue use: a context must be fully defined before start of injected conversion: all channels configured consecutively for the same ADC instance. Therefore, Number of calls of HAL\_ADCEx\_InjectedConfigChannel() must correspond to value of parameter InjectedNbrOfConversion for each context. Example 1: If 1 context intended to be used (or not use of this feature: QueueInjectedContext=DISABLE) and usage of the 3 first injected ranks (InjectedNbrOfConversion=3), HAL\_ADCEx\_InjectedConfigChannel() must be called once for each channel (3 times) before launching a conversion. This function must not be called to configure the 4th injected channel: it would start a new context into context queue.Example 2: If 2 contexts intended to be used and usage of the 3 first injected ranks (InjectedNbrOfConversion=3), HAL\_ADCEx\_InjectedConfigChannel() must be called once

for each channel and for each context (3 channels x 2 contexts = 6 calls). Conversion can start once the 1st context is set. The 2nd context can be set on the fly.

### **HAL\_ADCEx\_MultiModeConfigChannel**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_ADCEx_MultiModeConfigChannel(ADC_HandleTypeDef * hadc, ADC_MultiModeTypeDef * multimode)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Enable ADC multimode and configure multimode parameters.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hadc:</b> ADC handle</li> <li>• <b>multimode:</b> Structure of ADC multimode configuration</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Notes                | <ul style="list-style-type: none"> <li>• Possibility to update parameters on the fly: This function initializes multimode parameters, following calls to this function can be used to reconfigure some parameters of structure "ADC_MultiModeTypeDef" on the fly, without resetting the ADCs (both ADCs of the common group). The setting of these parameters is conditioned to ADC state. For parameters constraints, see comments of structure "ADC_MultiModeTypeDef".</li> <li>• To change back configuration from multimode to single mode, ADC must be reset (using function HAL_ADC_Init() ).</li> </ul> |

## **7.3 ADCEx Firmware driver defines**

### **7.3.1 ADCEx**

#### ***ADC Extended Analog Watchdog Mode***

ADC\_ANALOGWATCHDOG\_NONE  
 ADC\_ANALOGWATCHDOG\_SINGLE\_REG  
 ADC\_ANALOGWATCHDOG\_SINGLE\_INJEC  
 ADC\_ANALOGWATCHDOG\_SINGLE\_REGINJEC  
 ADC\_ANALOGWATCHDOG\_ALL\_REG  
 ADC\_ANALOGWATCHDOG\_ALL\_INJEC  
 ADC\_ANALOGWATCHDOG\_ALL\_REGINJEC

#### ***ADC Extended Analog Watchdog Selection***

ADC\_ANALOGWATCHDOG\_1  
 ADC\_ANALOGWATCHDOG\_2  
 ADC\_ANALOGWATCHDOG\_3

#### ***ADC Extended Channels***

ADC\_CHANNEL\_1  
 ADC\_CHANNEL\_2  
 ADC\_CHANNEL\_3

ADC\_CHANNEL\_4  
ADC\_CHANNEL\_5  
ADC\_CHANNEL\_6  
ADC\_CHANNEL\_7  
ADC\_CHANNEL\_8  
ADC\_CHANNEL\_9  
ADC\_CHANNEL\_10  
ADC\_CHANNEL\_11  
ADC\_CHANNEL\_12  
ADC\_CHANNEL\_13  
ADC\_CHANNEL\_14  
ADC\_CHANNEL\_15  
ADC\_CHANNEL\_16  
ADC\_CHANNEL\_17  
ADC\_CHANNEL\_18  
ADC\_CHANNEL\_VOPAMP1  
ADC\_CHANNEL\_TEMPSENSOR  
ADC\_CHANNEL\_VBAT  
ADC\_CHANNEL\_VOPAMP2  
ADC\_CHANNEL\_VOPAMP3  
ADC\_CHANNEL\_VOPAMP4  
ADC\_CHANNEL\_VREFINT

***ADC Extended Clock Prescaler***

|                          |                                                                           |
|--------------------------|---------------------------------------------------------------------------|
| ADC_CLOCK_ASYNC_DIV1     | ADC asynchronous clock derived from ADC dedicated PLL                     |
| ADC_CLOCK_SYNC_PCLK_DIV1 | ADC synchronous clock derived from AHB clock without prescaler            |
| ADC_CLOCK_SYNC_PCLK_DIV2 | ADC synchronous clock derived from AHB clock divided by a prescaler of 2U |
| ADC_CLOCK_SYNC_PCLK_DIV4 | ADC synchronous clock derived from AHB clock divided by a prescaler of 4U |

IS\_ADC\_CLOCKPRESCALER

***ADC Extended Dual ADC Mode***

ADC\_MODE\_INDEPENDENT  
ADC\_DUALMODE\_REGSIMULT\_INJECSIMULT  
ADC\_DUALMODE\_REGSIMULT.AlterTrig  
ADC\_DUALMODE\_REGINTERL\_INJECSIMULT  
ADC\_DUALMODE\_INJECSIMULT

ADC\_DUALMODE\_REGSIMULT

ADC\_DUALMODE\_INTERL

ADC\_DUALMODE\_ALTERTRIG

***ADC Extended Data Alignment***

ADC\_DATAALIGN\_RIGHT

ADC\_DATAALIGN\_LEFT

***ADC Extended Delay Between 2 Sampling Phases***

ADC\_TWOSAMPLINGDELAY\_1CYCLE

ADC\_TWOSAMPLINGDELAY\_2CYCLES

ADC\_TWOSAMPLINGDELAY\_3CYCLES

ADC\_TWOSAMPLINGDELAY\_4CYCLES

ADC\_TWOSAMPLINGDELAY\_5CYCLES

ADC\_TWOSAMPLINGDELAY\_6CYCLES

ADC\_TWOSAMPLINGDELAY\_7CYCLES

ADC\_TWOSAMPLINGDELAY\_8CYCLES

ADC\_TWOSAMPLINGDELAY\_9CYCLES

ADC\_TWOSAMPLINGDELAY\_10CYCLES

ADC\_TWOSAMPLINGDELAY\_11CYCLES

ADC\_TWOSAMPLINGDELAY\_12CYCLES

***ADC Extended DMA Mode for Dual ADC Mode***

ADC\_DMAACCESSMODE\_DISABLED DMA multimode disabled: each ADC will use its own DMA channel

ADC\_DMAACCESSMODE\_12\_10\_BITS DMA multimode enabled (one DMA channel for both ADC, DMA of ADC master) for 12 and 10 bits resolution

ADC\_DMAACCESSMODE\_8\_6\_BITS DMA multimode enabled (one DMA channel for both ADC, DMA of ADC master) for 8 and 6 bits resolution

***ADC Extended End of Regular Sequence/Conversion***

ADC\_EOC\_SINGLE\_CONV

ADC\_EOC\_SEQ\_CONV

***ADC Extended Error Code***

HAL\_ADC\_ERROR\_NONE No error

HAL\_ADC\_ERROR\_INTERNAL ADC IP internal error: if problem of clocking, enable/disable, erroneous state

HAL\_ADC\_ERROR\_OVR Overrun error

HAL\_ADC\_ERROR\_DMA DMA transfer error

HAL\_ADC\_ERROR\_JQOVF Injected context queue overflow error

***ADC Extended Event Type***

|                 |                                                                                             |
|-----------------|---------------------------------------------------------------------------------------------|
| ADC_AWD1_EVENT  | ADC Analog watchdog 1 event (main analog watchdog, present on all STM32 devices)            |
| ADC_AWD2_EVENT  | ADC Analog watchdog 2 event (additional analog watchdog, not present on all STM32 families) |
| ADC_AWD3_EVENT  | ADC Analog watchdog 3 event (additional analog watchdog, not present on all STM32 families) |
| ADC_OVR_EVENT   | ADC overrun event                                                                           |
| ADC_JQOVF_EVENT | ADC Injected Context Queue Overflow event                                                   |
| ADC_AWD_EVENT   |                                                                                             |

***ADCEx Exported Macros*****`_HAL_ADC_ENABLE`****Description:**

- Enable the ADC peripheral.

**Parameters:**

- `_HANDLE_`: ADC handle

**Return value:**

- None

**Notes:**

- ADC enable requires a delay for ADC stabilization time (refer to device datasheet, parameter tSTAB) On STM32F3 devices, some hardware constraints must be strictly respected before using this macro: ADC internal voltage regulator must be preliminarily enabled. This is performed by function `HAL_ADC_Init()`. ADC state requirements: ADC must be disabled, no conversion on going, no calibration on going. These checks are performed by functions `HAL_ADC_start_xxx()`.

**`_HAL_ADC_DISABLE`****Description:**

- Disable the ADC peripheral.

**Parameters:**

- `_HANDLE_`: ADC handle

**Return value:**

- None

**Notes:**

- On STM32F3 devices, some hardware constraints must be strictly respected before using this macro: ADC state requirements: ADC must be enabled, no conversion on going. These checks are performed by functions

HAL\_ADC\_start\_xxx().

### \_HAL\_ADC\_ENABLE\_IT

#### Description:

- Enable the ADC end of conversion interrupt.

#### Parameters:

- \_HANDLE\_: ADC handle
- \_INTERRUPT\_: ADC Interrupt This parameter can be any combination of the following values:
  - ADC\_IT\_RDY: ADC Ready (ADRDY) interrupt source
  - ADC\_IT\_EOSMP: ADC End of Sampling interrupt source
  - ADC\_IT\_EOC: ADC End of Regular Conversion interrupt source
  - ADC\_IT\_EOS: ADC End of Regular sequence of Conversions interrupt source
  - ADC\_IT\_OVR: ADC overrun interrupt source
  - ADC\_IT\_JEOC: ADC End of Injected Conversion interrupt source
  - ADC\_IT\_JEOS: ADC End of Injected sequence of Conversions interrupt source
  - ADC\_IT\_AWD1: ADC Analog watchdog 1 interrupt source (main analog watchdog, present on all STM32 devices)
  - ADC\_IT\_AWD2: ADC Analog watchdog 2 interrupt source (additional analog watchdog, present only on STM32F3 devices)
  - ADC\_IT\_AWD3: ADC Analog watchdog 3 interrupt source (additional analog watchdog, present only on STM32F3 devices)
  - ADC\_IT\_JQOVF: ADC Injected Context Queue Overflow interrupt source

#### Return value:

- None

### \_HAL\_ADC\_DISABLE\_IT

#### Description:

- Disable the ADC end of conversion interrupt.

#### Parameters:

- \_HANDLE\_: ADC handle
- \_INTERRUPT\_: ADC Interrupt This parameter can be any combination of the

following values:

- ADC\_IT\_RDY: ADC Ready (ADRDY) interrupt source
- ADC\_IT\_EOSMP: ADC End of Sampling interrupt source
- ADC\_IT\_EOC: ADC End of Regular Conversion interrupt source
- ADC\_IT\_EOS: ADC End of Regular sequence of Conversions interrupt source
- ADC\_IT\_OVR: ADC overrun interrupt source
- ADC\_IT\_JEOC: ADC End of Injected Conversion interrupt source
- ADC\_IT\_JEOS: ADC End of Injected sequence of Conversions interrupt source
- ADC\_IT\_AWD1: ADC Analog watchdog 1 interrupt source (main analog watchdog, present on all STM32 devices)
- ADC\_IT\_AWD2: ADC Analog watchdog 2 interrupt source (additional analog watchdog, present only on STM32F3 devices)
- ADC\_IT\_AWD3: ADC Analog watchdog 3 interrupt source (additional analog watchdog, present only on STM32F3 devices)
- ADC\_IT\_JQOVF: ADC Injected Context Queue Overflow interrupt source

#### Return value:

- None

### \_\_HAL\_ADC\_GET\_IT\_SOURCE

#### Description:

- Checks if the specified ADC interrupt source is enabled or disabled.

#### Parameters:

- \_\_HANDLE\_\_: ADC handle
- \_\_INTERRUPT\_\_: ADC interrupt source to check This parameter can be any combination of the following values:
  - ADC\_IT\_RDY: ADC Ready (ADRDY) interrupt source
  - ADC\_IT\_EOSMP: ADC End of Sampling interrupt source
  - ADC\_IT\_EOC: ADC End of Regular Conversion interrupt source
  - ADC\_IT\_EOS: ADC End of Regular sequence of Conversions interrupt source

- ADC\_IT\_OVR: ADC overrun interrupt source
- ADC\_IT\_JEOC: ADC End of Injected Conversion interrupt source
- ADC\_IT\_JEOS: ADC End of Injected sequence of Conversions interrupt source
- ADC\_IT\_AWD1: ADC Analog watchdog 1 interrupt source (main analog watchdog, present on all STM32 devices)
- ADC\_IT\_AWD2: ADC Analog watchdog 2 interrupt source (additional analog watchdog, present only on STM32F3 devices)
- ADC\_IT\_AWD3: ADC Analog watchdog 3 interrupt source (additional analog watchdog, present only on STM32F3 devices)
- ADC\_IT\_JQOVF: ADC Injected Context Queue Overflow interrupt source

**Return value:**

- State: of interruption (SET or RESET)

**\_HAL\_ADC\_GET\_FLAG****Description:**

- Get the selected ADC's flag status.

**Parameters:**

- HANDLE: ADC handle
- FLAG: ADC flag This parameter can be any combination of the following values:
  - ADC\_FLAG\_RDY: ADC Ready (ADRDY) flag
  - ADC\_FLAG\_EOSMP: ADC End of Sampling flag
  - ADC\_FLAG\_EOC: ADC End of Regular Conversion flag
  - ADC\_FLAG\_EOS: ADC End of Regular sequence of Conversions flag
  - ADC\_FLAG\_OVR: ADC overrun flag
  - ADC\_FLAG\_JEOC: ADC End of Injected Conversion flag
  - ADC\_FLAG\_JEOS: ADC End of Injected sequence of Conversions flag
  - ADC\_FLAG\_AWD1: ADC Analog watchdog 1 flag (main analog watchdog, present on all STM32 devices)
  - ADC\_FLAG\_AWD2: ADC Analog watchdog 2 flag (additional analog watchdog, present only on STM32F3 devices)

- ADC\_FLAG\_AWD3: ADC Analog watchdog 3 flag (additional analog watchdog, present only on STM32F3 devices)
- ADC\_FLAG\_JQOVF: ADC Injected Context Queue Overflow flag

**Return value:**

- None

[\\_\\_HAL\\_ADC\\_CLEAR\\_FLAG](#)**Description:**

- Clear the ADC's pending flags.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): ADC handle
- [\\_\\_FLAG\\_\\_](#): ADC flag This parameter can be any combination of the following values:
  - ADC\_FLAG\_RDY: ADC Ready (ADRDY) flag
  - ADC\_FLAG\_EOSMP: ADC End of Sampling flag
  - ADC\_FLAG\_EOC: ADC End of Regular Conversion flag
  - ADC\_FLAG\_EOS: ADC End of Regular sequence of Conversions flag
  - ADC\_FLAG\_OVR: ADC overrun flag
  - ADC\_FLAG\_JEOC: ADC End of Injected Conversion flag
  - ADC\_FLAG\_JEOS: ADC End of Injected sequence of Conversions flag
  - ADC\_FLAG\_AWD1: ADC Analog watchdog 1 flag (main analog watchdog, present on all STM32 devices)
  - ADC\_FLAG\_AWD2: ADC Analog watchdog 2 flag (additional analog watchdog, present only on STM32F3 devices)
  - ADC\_FLAG\_AWD3: ADC Analog watchdog 3 flag (additional analog watchdog, present only on STM32F3 devices)
  - ADC\_FLAG\_JQOVF: ADC Injected Context Queue Overflow flag

**Return value:**

- None

[\\_\\_HAL\\_ADC\\_RESET\\_HANDLE\\_STATE](#)**Description:**

- Reset ADC handle state.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): ADC handle

**Return value:**

- None

***External Trigger Edge of Injected Group***

ADC\_EXTERNALTRIGINJECCONV\_EDGE\_NONE  
ADC\_EXTERNALTRIGINJECCONV\_EDGE\_RISING  
ADC\_EXTERNALTRIGINJECCONV\_EDGE\_FALLING  
ADC\_EXTERNALTRIGINJECCONV\_EDGE\_RISINGFALLING

***ADC Extended External trigger enable and polarity selection for regular group***

ADC\_EXTERNALTRIGCONVEDGE\_NONE  
ADC\_EXTERNALTRIGCONVEDGE\_RISING  
ADC\_EXTERNALTRIGCONVEDGE\_FALLING  
ADC\_EXTERNALTRIGCONVEDGE\_RISINGFALLING

***External Trigger Source of Injected Group***

ADC\_EXTERNALTRIGINJECCONV\_T2\_CC1  
ADC\_EXTERNALTRIGINJECCONV\_T3\_CC1  
ADC\_EXTERNALTRIGINJECCONV\_T3\_CC3  
ADC\_EXTERNALTRIGINJECCONV\_T3\_CC4  
ADC\_EXTERNALTRIGINJECCONV\_T6\_TRGO  
ADC\_EXTERNALTRIGINJECCONV\_EXT\_IT15  
ADC\_EXTERNALTRIGINJECCONV\_T1\_CC3  
ADC\_EXTERNALTRIGINJECCONV\_T4\_CC3  
ADC\_EXTERNALTRIGINJECCONV\_T4\_CC4  
ADC\_EXTERNALTRIGINJECCONV\_T7\_TRGO  
ADC\_EXTERNALTRIGINJECCONV\_T8\_CC2  
ADC\_EXTERNALTRIGINJECCONV\_T1\_CC4  
ADC\_EXTERNALTRIGINJECCONV\_T1\_TRGO  
ADC\_EXTERNALTRIGINJECCONV\_T1\_TRGO2  
ADC\_EXTERNALTRIGINJECCONV\_T2\_TRGO  
ADC\_EXTERNALTRIGINJECCONV\_T3\_TRGO  
ADC\_EXTERNALTRIGINJECCONV\_T4\_TRGO  
ADC\_EXTERNALTRIGINJECCONV\_T8\_CC4  
ADC\_EXTERNALTRIGINJECCONV\_T8\_TRGO  
ADC\_EXTERNALTRIGINJECCONV\_T8\_TRGO2  
ADC\_EXTERNALTRIGINJECCONV\_T15\_TRGO  
ADC\_INJECTED\_SOFTWARE\_START  
IS\_ADC\_EXTTRIGINJEC

**IS\_ADC\_EXTRIGINJEC*****ADC Extended External trigger selection for regular group***

|                               |                                                                         |
|-------------------------------|-------------------------------------------------------------------------|
| ADC_EXTERNALTRIGCONV_T1_CC1   | < List of external triggers with generic trigger name, independently of |
| ADC_EXTERNALTRIGCONV_T1_CC2   |                                                                         |
| ADC_EXTERNALTRIGCONV_T2_CC2   |                                                                         |
| ADC_EXTERNALTRIGCONV_T3_CC4   |                                                                         |
| ADC_EXTERNALTRIGCONV_T4_CC4   |                                                                         |
| ADC_EXTERNALTRIGCONV_T6_TRGO  |                                                                         |
| ADC_EXTERNALTRIGCONV_EXT_IT11 | External triggers of regular group for ADC3&ADC4 only                   |
| ADC_EXTERNALTRIGCONV_T2_CC1   |                                                                         |
| ADC_EXTERNALTRIGCONV_T2_CC3   |                                                                         |
| ADC_EXTERNALTRIGCONV_T3_CC1   |                                                                         |
| ADC_EXTERNALTRIGCONV_T4_CC1   |                                                                         |
| ADC_EXTERNALTRIGCONV_T7_TRGO  |                                                                         |
| ADC_EXTERNALTRIGCONV_T8_CC1   |                                                                         |
| ADC_EXTERNALTRIGCONV_EXT_IT2  | External triggers of regular group for ADC1&ADC2, ADC3&ADC4             |
| ADC_EXTERNALTRIGCONV_T1_CC3   |                                                                         |
| ADC_EXTERNALTRIGCONV_T1_TRGO  |                                                                         |
| ADC_EXTERNALTRIGCONV_T1_TRGO2 |                                                                         |
| ADC_EXTERNALTRIGCONV_T2_TRGO  |                                                                         |
| ADC_EXTERNALTRIGCONV_T3_TRGO  |                                                                         |
| ADC_EXTERNALTRIGCONV_T4_TRGO  |                                                                         |
| ADC_EXTERNALTRIGCONV_T8_TRGO  |                                                                         |
| ADC_EXTERNALTRIGCONV_T8_TRGO2 |                                                                         |
| ADC_EXTERNALTRIGCONV_T15_TRGO |                                                                         |
| ADC_SOFTWARE_START            |                                                                         |

***ADC Extended Flags Definition***

|                |                                                  |
|----------------|--------------------------------------------------|
| ADC_FLAG_RDY   | ADC Ready (ADRDY) flag                           |
| ADC_FLAG_EOSMP | ADC End of Sampling flag                         |
| ADC_FLAG_EOC   | ADC End of Regular Conversion flag               |
| ADC_FLAG_EOS   | ADC End of Regular sequence of Conversions flag  |
| ADC_FLAG_OVR   | ADC overrun flag                                 |
| ADC_FLAG_JEOC  | ADC End of Injected Conversion flag              |
| ADC_FLAG_JEOS  | ADC End of Injected sequence of Conversions flag |

|                |                                                                                          |
|----------------|------------------------------------------------------------------------------------------|
| ADC_FLAG_AWD1  | ADC Analog watchdog 1 flag (main analog watchdog, present on all STM32 devices)          |
| ADC_FLAG_AWD2  | ADC Analog watchdog 2 flag (additional analog watchdog, present only on STM32F3 devices) |
| ADC_FLAG_AWD3  | ADC Analog watchdog 3 flag (additional analog watchdog, present only on STM32F3 devices) |
| ADC_FLAG_JQOVF | ADC Injected Context Queue Overflow flag                                                 |
| ADC_FLAG_AWD   |                                                                                          |

***ADC Extended Injected Channel Rank***

ADC\_INJECTED\_RANK\_1  
ADC\_INJECTED\_RANK\_2  
ADC\_INJECTED\_RANK\_3  
ADC\_INJECTED\_RANK\_4

***ADC Extended Internal HAL driver trigger selection for injected group***

ADC1\_2\_EXTERNALTRIGINJEC\_T1\_TRGO  
ADC1\_2\_EXTERNALTRIGINJEC\_T1\_CC4  
ADC1\_2\_EXTERNALTRIGINJEC\_T2\_TRGO  
ADC1\_2\_EXTERNALTRIGINJEC\_T2\_CC1  
ADC1\_2\_EXTERNALTRIGINJEC\_T3\_CC4  
ADC1\_2\_EXTERNALTRIGINJEC\_T4\_TRGO  
ADC1\_2\_EXTERNALTRIGINJEC\_EXT\_IT15  
ADC1\_2\_EXTERNALTRIGINJEC\_T8\_CC4  
ADC1\_2\_EXTERNALTRIGINJEC\_T1\_TRGO2  
ADC1\_2\_EXTERNALTRIGINJEC\_T8\_TRGO  
ADC1\_2\_EXTERNALTRIGINJEC\_T8\_TRGO2  
ADC1\_2\_EXTERNALTRIGINJEC\_T3\_CC3  
ADC1\_2\_EXTERNALTRIGINJEC\_T3\_TRGO  
ADC1\_2\_EXTERNALTRIGINJEC\_T3\_CC1  
ADC1\_2\_EXTERNALTRIGINJEC\_T6\_TRGO  
ADC1\_2\_EXTERNALTRIGINJEC\_T15\_TRGO  
ADC3\_4\_EXTERNALTRIGINJEC\_T1\_TRGO  
ADC3\_4\_EXTERNALTRIGINJEC\_T1\_CC4  
ADC3\_4\_EXTERNALTRIGINJEC\_T4\_CC3  
ADC3\_4\_EXTERNALTRIGINJEC\_T8\_CC2  
ADC3\_4\_EXTERNALTRIGINJEC\_T8\_CC4  
ADC3\_4\_EXTERNALTRIGINJEC\_T4\_CC4  
ADC3\_4\_EXTERNALTRIGINJEC\_T4\_TRGO

ADC3\_4\_EXTERNALTRIGINJEC\_T1\_TRGO2  
ADC3\_4\_EXTERNALTRIGINJEC\_T8\_TRGO  
ADC3\_4\_EXTERNALTRIGINJEC\_T8\_TRGO2  
ADC3\_4\_EXTERNALTRIGINJEC\_T1\_CC3  
ADC3\_4\_EXTERNALTRIGINJEC\_T3\_TRGO  
ADC3\_4\_EXTERNALTRIGINJEC\_T2\_TRGO  
ADC3\_4\_EXTERNALTRIGINJEC\_T7\_TRGO  
ADC3\_4\_EXTERNALTRIGINJEC\_T15\_TRGO

***ADC Extended Internal HAL driver trigger selection for regular group***

ADC1\_2\_EXTERNALTRIG\_T1\_CC1  
ADC1\_2\_EXTERNALTRIG\_T1\_CC2  
ADC1\_2\_EXTERNALTRIG\_T1\_CC3  
ADC1\_2\_EXTERNALTRIG\_T2\_CC2  
ADC1\_2\_EXTERNALTRIG\_T3\_TRGO  
ADC1\_2\_EXTERNALTRIG\_T4\_CC4  
ADC1\_2\_EXTERNALTRIG\_EXT\_IT11  
ADC1\_2\_EXTERNALTRIG\_T8\_TRGO  
ADC1\_2\_EXTERNALTRIG\_T8\_TRGO2  
ADC1\_2\_EXTERNALTRIG\_T1\_TRGO  
ADC1\_2\_EXTERNALTRIG\_T1\_TRGO2  
ADC1\_2\_EXTERNALTRIG\_T2\_TRGO  
ADC1\_2\_EXTERNALTRIG\_T4\_TRGO  
ADC1\_2\_EXTERNALTRIG\_T6\_TRGO  
ADC1\_2\_EXTERNALTRIG\_T15\_TRGO  
ADC1\_2\_EXTERNALTRIG\_T3\_CC4  
ADC3\_4\_EXTERNALTRIG\_T3\_CC1  
ADC3\_4\_EXTERNALTRIG\_T2\_CC3  
ADC3\_4\_EXTERNALTRIG\_T1\_CC3  
ADC3\_4\_EXTERNALTRIG\_T8\_CC1  
ADC3\_4\_EXTERNALTRIG\_T8\_TRGO  
ADC3\_4\_EXTERNALTRIG\_EXT\_IT2  
ADC3\_4\_EXTERNALTRIG\_T4\_CC1  
ADC3\_4\_EXTERNALTRIG\_T2\_TRGO  
ADC3\_4\_EXTERNALTRIG\_T8\_TRGO2  
ADC3\_4\_EXTERNALTRIG\_T1\_TRGO  
ADC3\_4\_EXTERNALTRIG\_T1\_TRGO2

ADC3\_4\_EXTERNALTRIG\_T3\_TRGO  
ADC3\_4\_EXTERNALTRIG\_T4\_TRGO  
ADC3\_4\_EXTERNALTRIG\_T7\_TRGO  
ADC3\_4\_EXTERNALTRIG\_T15\_TRGO  
ADC3\_4\_EXTERNALTRIG\_T2\_CC1

***ADC Extended Interrupts Definition***

|              |                                                                                                      |
|--------------|------------------------------------------------------------------------------------------------------|
| ADC_IT_RDY   | ADC Ready (ADRDY) interrupt source                                                                   |
| ADC_IT_EOSMP | ADC End of Sampling interrupt source                                                                 |
| ADC_IT_EOC   | ADC End of Regular Conversion interrupt source                                                       |
| ADC_IT_EOS   | ADC End of Regular sequence of Conversions interrupt source                                          |
| ADC_IT_OVR   | ADC overrun interrupt source                                                                         |
| ADC_IT_JEOC  | ADC End of Injected Conversion interrupt source                                                      |
| ADC_IT_JEOS  | ADC End of Injected sequence of Conversions interrupt source                                         |
| ADC_IT_AWD1  | ADC Analog watchdog 1 interrupt source (main analog watchdog, present on all STM32 devices)          |
| ADC_IT_AWD2  | ADC Analog watchdog 2 interrupt source (additional analog watchdog, present only on STM32F3 devices) |
| ADC_IT_AWD3  | ADC Analog watchdog 3 interrupt source (additional analog watchdog, present only on STM32F3 devices) |
| ADC_IT_JQOVF | ADC Injected Context Queue Overflow interrupt source                                                 |
| ADC_IT_AWD   |                                                                                                      |

***ADC Extended Offset Number***

ADC\_OFFSET\_NONE  
ADC\_OFFSET\_1  
ADC\_OFFSET\_2  
ADC\_OFFSET\_3  
ADC\_OFFSET\_4

***ADC Extended overrun***

ADC\_OVR\_DATA\_OVERWRITTEN Default setting, to be used for compatibility with other STM32 devices

ADC\_OVR\_DATA\_PRESERVED

***ADC Extended Range Verification***

IS\_ADC\_RANGE

***ADC Extended rank into regular group***

ADC\_REGULAR\_RANK\_1  
ADC\_REGULAR\_RANK\_2  
ADC\_REGULAR\_RANK\_3  
ADC\_REGULAR\_RANK\_4

ADC\_REGULAR\_RANK\_5  
ADC\_REGULAR\_RANK\_6  
ADC\_REGULAR\_RANK\_7  
ADC\_REGULAR\_RANK\_8  
ADC\_REGULAR\_RANK\_9  
ADC\_REGULAR\_RANK\_10  
ADC\_REGULAR\_RANK\_11  
ADC\_REGULAR\_RANK\_12  
ADC\_REGULAR\_RANK\_13  
ADC\_REGULAR\_RANK\_14  
ADC\_REGULAR\_RANK\_15  
ADC\_REGULAR\_RANK\_16

***ADC Extended Resolution***

ADC\_RESOLUTION\_12B ADC 12-bit resolution  
ADC\_RESOLUTION\_10B ADC 10-bit resolution  
ADC\_RESOLUTION\_8B ADC 8-bit resolution  
ADC\_RESOLUTION\_6B ADC 6-bit resolution

***ADC Extended Sampling Times***

ADC\_SAMPLETIME\_1CYCLE\_5 Sampling time 1.5 ADC clock cycle  
ADC\_SAMPLETIME\_2CYCLES\_5 Sampling time 2.5 ADC clock cycles  
ADC\_SAMPLETIME\_4CYCLES\_5 Sampling time 4.5 ADC clock cycles  
ADC\_SAMPLETIME\_7CYCLES\_5 Sampling time 7.5 ADC clock cycles  
ADC\_SAMPLETIME\_19CYCLES\_5 Sampling time 19.5 ADC clock cycles  
ADC\_SAMPLETIME\_61CYCLES\_5 Sampling time 61.5 ADC clock cycles  
ADC\_SAMPLETIME\_181CYCLES\_5 Sampling time 181.5 ADC clock cycles  
ADC\_SAMPLETIME\_601CYCLES\_5 Sampling time 601.5 ADC clock cycles

***ADC Extended Scan Mode***

ADC\_SCAN\_DISABLE  
ADC\_SCAN\_ENABLE

***ADC Extended Single-ended/Differential input mode***

ADC\_SINGLE\_ENDED  
ADC\_DIFFERENTIAL\_ENDED

## 8 HAL CAN Generic Driver

### 8.1 CAN Firmware driver registers structures

#### 8.1.1 CAN\_InitTypeDef

##### Data Fields

- *uint32\_t Prescaler*
- *uint32\_t Mode*
- *uint32\_t SJW*
- *uint32\_t BS1*
- *uint32\_t BS2*
- *uint32\_t TTCM*
- *uint32\_t ABOM*
- *uint32\_t AWUM*
- *uint32\_t NART*
- *uint32\_t RFLM*
- *uint32\_t TXFP*

##### Field Documentation

- ***uint32\_t CAN\_InitTypeDef::Prescaler***  
Specifies the length of a time quantum. This parameter must be a number between Min\_Data = 1 and Max\_Data = 1024.
- ***uint32\_t CAN\_InitTypeDef::Mode***  
Specifies the CAN operating mode. This parameter can be a value of [\*CAN\\_operating\\_mode\*](#)
- ***uint32\_t CAN\_InitTypeDef::SJW***  
Specifies the maximum number of time quanta the CAN hardware is allowed to lengthen or shorten a bit to perform resynchronization. This parameter can be a value of [\*CAN\\_synchronisation\\_jump\\_width\*](#)
- ***uint32\_t CAN\_InitTypeDef::BS1***  
Specifies the number of time quanta in Bit Segment 1. This parameter can be a value of [\*CAN\\_time\\_quantum\\_in\\_bit\\_segment\\_1\*](#)
- ***uint32\_t CAN\_InitTypeDef::BS2***  
Specifies the number of time quanta in Bit Segment 2. This parameter can be a value of [\*CAN\\_time\\_quantum\\_in\\_bit\\_segment\\_2\*](#)
- ***uint32\_t CAN\_InitTypeDef::TTCM***  
Enable or disable the time triggered communication mode. This parameter can be set to ENABLE or DISABLE.
- ***uint32\_t CAN\_InitTypeDef::ABOM***  
Enable or disable the automatic bus-off management. This parameter can be set to ENABLE or DISABLE.
- ***uint32\_t CAN\_InitTypeDef::AWUM***  
Enable or disable the automatic wake-up mode. This parameter can be set to ENABLE or DISABLE.
- ***uint32\_t CAN\_InitTypeDef::NART***  
Enable or disable the non-automatic retransmission mode. This parameter can be set to ENABLE or DISABLE.
- ***uint32\_t CAN\_InitTypeDef::RFLM***  
Enable or disable the Receive FIFO Locked mode. This parameter can be set to ENABLE or DISABLE.

- ***uint32\_t CAN\_InitTypeDef::TXFP***  
Enable or disable the transmit FIFO priority. This parameter can be set to ENABLE or DISABLE.

## 8.1.2 CAN\_FilterConfTypeDef

### Data Fields

- ***uint32\_t FilterIdHigh***
- ***uint32\_t FilterIdLow***
- ***uint32\_t FilterMaskIdHigh***
- ***uint32\_t FilterMaskIdLow***
- ***uint32\_t FilterFIFOAssignment***
- ***uint32\_t FilterNumber***
- ***uint32\_t FilterMode***
- ***uint32\_t FilterScale***
- ***uint32\_t FilterActivation***
- ***uint32\_t BankNumber***

### Field Documentation

- ***uint32\_t CAN\_FilterConfTypeDef::FilterIdHigh***  
Specifies the filter identification number (MSBs for a 32-bit configuration, first one for a 16-bit configuration). This parameter must be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFF.
- ***uint32\_t CAN\_FilterConfTypeDef::FilterIdLow***  
Specifies the filter identification number (LSBs for a 32-bit configuration, second one for a 16-bit configuration). This parameter must be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFF.
- ***uint32\_t CAN\_FilterConfTypeDef::FilterMaskIdHigh***  
Specifies the filter mask number or identification number, according to the mode (MSBs for a 32-bit configuration, first one for a 16-bit configuration). This parameter must be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFF.
- ***uint32\_t CAN\_FilterConfTypeDef::FilterMaskIdLow***  
Specifies the filter mask number or identification number, according to the mode (LSBs for a 32-bit configuration, second one for a 16-bit configuration). This parameter must be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFF.
- ***uint32\_t CAN\_FilterConfTypeDef::FilterFIFOAssignment***  
Specifies the FIFO (0 or 1U) which will be assigned to the filter. This parameter can be a value of [CAN\\_filter\\_FIFO](#)
- ***uint32\_t CAN\_FilterConfTypeDef::FilterNumber***  
Specifies the filter which will be initialized. This parameter must be a number between Min\_Data = 0 and Max\_Data = 27.
- ***uint32\_t CAN\_FilterConfTypeDef::FilterMode***  
Specifies the filter mode to be initialized. This parameter can be a value of [CAN\\_filter\\_mode](#)
- ***uint32\_t CAN\_FilterConfTypeDef::FilterScale***  
Specifies the filter scale. This parameter can be a value of [CAN\\_filter\\_scale](#)
- ***uint32\_t CAN\_FilterConfTypeDef::FilterActivation***  
Enable or disable the filter. This parameter can be set to ENABLE or DISABLE.
- ***uint32\_t CAN\_FilterConfTypeDef::BankNumber***  
Select the start slave bank filter F3 devices don't support CAN2 interface (Slave). Therefore this parameter is meaningless but it has been kept for compatibility accross STM32 families

### 8.1.3 CanTxMsgTypeDef

#### Data Fields

- *uint32\_t StdId*
- *uint32\_t ExtId*
- *uint32\_t IDE*
- *uint32\_t RTR*
- *uint32\_t DLC*
- *uint8\_t Data*

#### Field Documentation

- ***uint32\_t CanTxMsgTypeDef::StdId***  
Specifies the standard identifier. This parameter must be a number between Min\_Data = 0 and Max\_Data = 0x7FF.
- ***uint32\_t CanTxMsgTypeDef::ExtId***  
Specifies the extended identifier. This parameter must be a number between Min\_Data = 0 and Max\_Data = 0x1FFFFFFF.
- ***uint32\_t CanTxMsgTypeDef::IDE***  
Specifies the type of identifier for the message that will be transmitted. This parameter can be a value of [CAN\\_identifier\\_type](#)
- ***uint32\_t CanTxMsgTypeDef::RTR***  
Specifies the type of frame for the message that will be transmitted. This parameter can be a value of [CAN\\_remote\\_transmission\\_request](#)
- ***uint32\_t CanTxMsgTypeDef::DLC***  
Specifies the length of the frame that will be transmitted. This parameter must be a number between Min\_Data = 0 and Max\_Data = 8.
- ***uint8\_t CanTxMsgTypeDef::Data[8]***  
Contains the data to be transmitted. This parameter must be a number between Min\_Data = 0 and Max\_Data = 0xFF.

### 8.1.4 CanRxMsgTypeDef

#### Data Fields

- *uint32\_t StdId*
- *uint32\_t ExtId*
- *uint32\_t IDE*
- *uint32\_t RTR*
- *uint32\_t DLC*
- *uint8\_t Data*
- *uint32\_t FMI*
- *uint32\_t FIFONumber*

#### Field Documentation

- ***uint32\_t CanRxMsgTypeDef::StdId***  
Specifies the standard identifier. This parameter must be a number between Min\_Data = 0 and Max\_Data = 0x7FF.
- ***uint32\_t CanRxMsgTypeDef::ExtId***  
Specifies the extended identifier. This parameter must be a number between Min\_Data = 0 and Max\_Data = 0x1FFFFFFF.
- ***uint32\_t CanRxMsgTypeDef::IDE***  
Specifies the type of identifier for the message that will be received. This parameter can be a value of [CAN\\_identifier\\_type](#)

- ***uint32\_t CanRxMsgTypeDef::RTR***  
Specifies the type of frame for the received message. This parameter can be a value of **CAN\_remote\_transmission\_request**
- ***uint32\_t CanRxMsgTypeDef::DLC***  
Specifies the length of the frame that will be received. This parameter must be a number between Min\_Data = 0 and Max\_Data = 8.
- ***uint8\_t CanRxMsgTypeDef::Data[8]***  
Contains the data to be received. This parameter must be a number between Min\_Data = 0 and Max\_Data = 0xFF.
- ***uint32\_t CanRxMsgTypeDef::FMI***  
Specifies the index of the filter the message stored in the mailbox passes through. This parameter must be a number between Min\_Data = 0 and Max\_Data = 0xFF.
- ***uint32\_t CanRxMsgTypeDef::FIFONumber***  
Specifies the receive FIFO number. This parameter can be CAN\_FIFO0 or CAN\_FIFO1

## 8.1.5 CAN\_HandleTypeDef

### Data Fields

- ***CAN\_TypeDef \* Instance***
- ***CAN\_InitTypeDef Init***
- ***CanTxMsgTypeDef \* pTxMsg***
- ***CanRxMsgTypeDef \* pRxMsg***
- ***CanRxMsgTypeDef \* pRx1Msg***
- ***HAL\_LockTypeDef Lock***
- ***\_\_IO HAL\_CAN\_StateTypeDef State***
- ***\_\_IO uint32\_t ErrorCode***

### Field Documentation

- ***CAN\_TypeDef\* CAN\_HandleTypeDef::Instance***  
Register base address
- ***CAN\_InitTypeDef CAN\_HandleTypeDef::Init***  
CAN required parameters
- ***CanTxMsgTypeDef\* CAN\_HandleTypeDef::pTxMsg***  
Pointer to transmit structure
- ***CanRxMsgTypeDef\* CAN\_HandleTypeDef::pRxMsg***  
Pointer to reception structure for RX FIFO0 msg
- ***CanRxMsgTypeDef\* CAN\_HandleTypeDef::pRx1Msg***  
Pointer to reception structure for RX FIFO1 msg
- ***HAL\_LockTypeDef CAN\_HandleTypeDef::Lock***  
CAN locking object
- ***\_\_IO HAL\_CAN\_StateTypeDef CAN\_HandleTypeDef::State***  
CAN communication state
- ***\_\_IO uint32\_t CAN\_HandleTypeDef::ErrorCode***  
CAN Error code This parameter can be a value of **CAN\_Error\_Code**

## 8.2 CAN Firmware driver API description

### 8.2.1 How to use this driver

1. Enable the CAN controller interface clock using `__HAL_RCC_CAN1_CLK_ENABLE();`
2. CAN pins configuration
  - Enable the clock for the CAN GPIOs using the following function:  
`__HAL_RCC_GPIOx_CLK_ENABLE();`

- Connect and configure the involved CAN pins to AF9 using the following function `HAL_GPIO_Init()`;
- 3. Initialise and configure the CAN using `HAL_CAN_Init()` function.
- 4. Transmit the desired CAN frame using `HAL_CAN_Transmit()` function.
- 5. Or transmit the desired CAN frame using `HAL_CAN_Transmit_IT()` function.
- 6. Receive a CAN frame using `HAL_CAN_Receive()` function.
- 7. Or receive a CAN frame using `HAL_CAN_Receive_IT()` function.

### Polling mode IO operation

- Start the CAN peripheral transmission and wait the end of this operation using `HAL_CAN_Transmit()`, at this stage user can specify the value of timeout according to his end application
- Start the CAN peripheral reception and wait the end of this operation using `HAL_CAN_Receive()`, at this stage user can specify the value of timeout according to his end application

### Interrupt mode IO operation

- Start the CAN peripheral transmission using `HAL_CAN_Transmit_IT()`
- Start the CAN peripheral reception using `HAL_CAN_Receive_IT()`
- Use `HAL_CAN_IRQHandler()` called under the used CAN Interrupt subroutine
- At CAN end of transmission `HAL_CAN_TxCpltCallback()` function is executed and user can add his own code by customization of function pointer `HAL_CAN_TxCpltCallback`
- In case of CAN Error, `HAL_CAN_ErrorCallback()` function is executed and user can add his own code by customization of function pointer `HAL_CAN_ErrorCallback`

### CAN HAL driver macros list

Below the list of most used macros in CAN HAL driver.

- `_HAL_CAN_ENABLE_IT`: Enable the specified CAN interrupts
- `_HAL_CAN_DISABLE_IT`: Disable the specified CAN interrupts
- `_HAL_CAN_GET_IT_SOURCE`: Check if the specified CAN interrupt source is enabled or disabled
- `_HAL_CAN_CLEAR_FLAG`: Clear the CAN's pending flags
- `_HAL_CAN_GET_FLAG`: Get the selected CAN's flag status



You can refer to the CAN HAL driver header file for more useful macros

## 8.2.2 Initialization and de-initialization functions

This section provides functions allowing to:

- Initialize and configure the CAN.
- De-initialize the CAN.

This section contains the following APIs:

- [`HAL\_CAN\_Init\(\)`](#)
- [`HAL\_CAN\_ConfigFilter\(\)`](#)
- [`HAL\_CAN\_DeInit\(\)`](#)
- [`HAL\_CAN\_MspInit\(\)`](#)

- [\*HAL\\_CAN\\_MspDeInit\(\)\*](#)

### 8.2.3 Peripheral State and Error functions

This subsection provides functions allowing to :

- Check the CAN state.
- Check CAN Errors detected during interrupt process

This section contains the following APIs:

- [\*HAL\\_CAN\\_GetState\(\)\*](#)
- [\*HAL\\_CAN\\_GetError\(\)\*](#)

### 8.2.4 Detailed description of functions

#### **HAL\_CAN\_Init**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CAN_Init (CAN_HandleTypeDef * hcan)</b>                                                                                                             |
| Function description | Initializes the CAN peripheral according to the specified parameters in the CAN_InitStruct.                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                       |

#### **HAL\_CAN\_ConfigFilter**

|                      |                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CAN_ConfigFilter (CAN_HandleTypeDef * hcan, CAN_FilterConfTypeDef * sFilterConfig)</b>                                                                                                                                                                                                |
| Function description | Configures the CAN reception filter according to the specified parameters in the CAN_FilterInitStruct.                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> <li>• <b>sFilterConfig:</b> pointer to a CAN_FilterConfTypeDef structure that contains the filter configuration information.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                |

#### **HAL\_CAN\_DeInit**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CAN_DeInit (CAN_HandleTypeDef * hcan)</b>                                                                                                           |
| Function description | Deinitializes the CANx peripheral registers to their default reset values.                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                       |

**HAL\_CAN\_MspInit**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_CAN_MspInit (CAN_HandleTypeDef * hcan)</b>                                                                                                                     |
| Function description | Initializes the CAN MSP.                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                              |

**HAL\_CAN\_MspDelInit**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_CAN_MspDelInit (CAN_HandleTypeDef * hcan)</b>                                                                                                                  |
| Function description | Deinitializes the CAN MSP.                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                              |

**HAL\_CAN\_Transmit**

|                      |                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CAN_Transmit (CAN_HandleTypeDef * hcan, uint32_t Timeout)</b>                                                                                                                                 |
| Function description | Initiates and transmits a CAN frame message.                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li><li>• <b>Timeout:</b> Timeout duration.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                                                                   |

**HAL\_CAN\_Transmit\_IT**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CAN_Transmit_IT (CAN_HandleTypeDef * hcan)</b>                                                                                                    |
| Function description | Initiates and transmits a CAN frame message.                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                       |

**HAL\_CAN\_Receive**

|                      |                                                                                                                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CAN_Receive (CAN_HandleTypeDef * hcan, uint8_t FIFONumber, uint32_t Timeout)</b>                                                                                                                                                        |
| Function description | Receives a correct CAN frame.                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li><li>• <b>FIFONumber:</b> FIFO number.</li><li>• <b>Timeout:</b> Timeout duration.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                                                                                                             |

**HAL\_CAN\_Receive\_IT**

|                      |                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CAN_Receive_IT<br/>(CAN_HandleTypeDef * hcan, uint8_t FIFONumber)</b>                                                                                                                          |
| Function description | Receives a correct CAN frame.                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> <li>• <b>FIFONumber:</b> FIFO number.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                  |

**HAL\_CAN\_Sleep**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CAN_Sleep (CAN_HandleTypeDef * hcan)</b>                                                                                                            |
| Function description | Enters the Sleep (low power) mode.                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status.</li> </ul>                                                                                                      |

**HAL\_CAN\_WakeUp**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CAN_WakeUp<br/>(CAN_HandleTypeDef * hcan)</b>                                                                                                       |
| Function description | Wakes up the CAN peripheral from sleep mode, after that the CAN peripheral is in the normal mode.                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status.</li> </ul>                                                                                                      |

**HAL\_CAN\_IRQHandler**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_CAN_IRQHandler (CAN_HandleTypeDef * hcan)</b>                                                                                                                    |
| Function description | Handles CAN interrupt request.                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_CAN\_TxCpltCallback**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_CAN_TxCpltCallback (CAN_HandleTypeDef * hcan)</b>                                                                                                                |
| Function description | Transmission complete callback in non blocking mode.                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_CAN\_RxCpltCallback**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_CAN_RxCpltCallback (CAN_HandleTypeDef * hcan)</b>                                                                                                                |
| Function description | Transmission complete callback in non blocking mode.                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_CAN\_ErrorCallback**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_CAN_ErrorCallback (CAN_HandleTypeDef * hcan)</b>                                                                                                                 |
| Function description | Error CAN callback.                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_CAN\_GetError**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_CAN_GetError (CAN_HandleTypeDef * hcan)</b>                                                                                                                  |
| Function description | Return the CAN error code.                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>CAN:</b> Error Code</li> </ul>                                                                                                   |

**HAL\_CAN\_GetState**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_CAN_StateTypeDef HAL_CAN_GetState (CAN_HandleTypeDef * hcan)</b>                                                                                                      |
| Function description | return the CAN state                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcan:</b> pointer to a CAN_HandleTypeDef structure that contains the configuration information for the specified CAN.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                        |

## 8.3 CAN Firmware driver defines

### 8.3.1 CAN

**CAN Error Code**

|                           |             |
|---------------------------|-------------|
| <b>HAL_CAN_ERROR_NONE</b> | No error    |
| <b>HAL_CAN_ERROR_EWG</b>  | EWG error   |
| <b>HAL_CAN_ERROR_EPV</b>  | EPV error   |
| <b>HAL_CAN_ERROR_BOF</b>  | BOF error   |
| <b>HAL_CAN_ERROR_STF</b>  | Stuff error |
| <b>HAL_CAN_ERROR_FOR</b>  | Form error  |

|                      |                      |
|----------------------|----------------------|
| HAL_CAN_ERROR_ACK    | Acknowledgment error |
| HAL_CAN_ERROR_BR     | Bit recessive        |
| HAL_CAN_ERROR_BD     | LEC dominant         |
| HAL_CAN_ERROR_CRC    | LEC transfer error   |
| HAL_CAN_ERROR_FOV0   | FIFO0 overrun error  |
| HAL_CAN_ERROR_FOV1   | FIFO1 overrun error  |
| HAL_CAN_ERROR_TXFAIL | Transmit failure     |

**CAN Exported Macros**

|                                          |                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>_HAL_CAN_RESET_HANDLE_STATE</code> | <b>Description:</b> <ul style="list-style-type: none"><li>Reset CAN handle state.</li></ul> <b>Parameters:</b> <ul style="list-style-type: none"><li><code>_HANDLE_</code>: CAN handle.</li></ul> <b>Return value:</b> <ul style="list-style-type: none"><li>None</li></ul>                                                                                                                                    |
| <code>_HAL_CAN_ENABLE_IT</code>          | <b>Description:</b> <ul style="list-style-type: none"><li>Enable the specified CAN interrupts.</li></ul> <b>Parameters:</b> <ul style="list-style-type: none"><li><code>_HANDLE_</code>: CAN handle.</li><li><code>_INTERRUPT_</code>: CAN Interrupt</li></ul> <b>Return value:</b> <ul style="list-style-type: none"><li>None</li></ul>                                                                       |
| <code>_HAL_CAN_DISABLE_IT</code>         | <b>Description:</b> <ul style="list-style-type: none"><li>Disable the specified CAN interrupts.</li></ul> <b>Parameters:</b> <ul style="list-style-type: none"><li><code>_HANDLE_</code>: CAN handle.</li><li><code>_INTERRUPT_</code>: CAN Interrupt</li></ul> <b>Return value:</b> <ul style="list-style-type: none"><li>None</li></ul>                                                                      |
| <code>_HAL_CAN_MSG_PENDING</code>        | <b>Description:</b> <ul style="list-style-type: none"><li>Return the number of pending received messages.</li></ul> <b>Parameters:</b> <ul style="list-style-type: none"><li><code>_HANDLE_</code>: CAN handle.</li><li><code>_FIFONUMBER_</code>: Receive FIFO number, CAN_FIFO0 or CAN_FIFO1.</li></ul> <b>Return value:</b> <ul style="list-style-type: none"><li>The: number of pending message.</li></ul> |

[\\_\\_HAL\\_CAN\\_GET\\_FLAG](#)**Description:**

- Check whether the specified CAN flag is set or not.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the CAN Handle.
- [\\_\\_FLAG\\_\\_](#): specifies the flag to check.  
This parameter can be one of the following values:
  - CAN\_TSR\_RQCP0: Request MailBox0 Flag
  - CAN\_TSR\_RQCP1: Request MailBox1 Flag
  - CAN\_TSR\_RQCP2: Request MailBox2 Flag
  - CAN\_FLAG\_TXOK0: Transmission OK MailBox0 Flag
  - CAN\_FLAG\_TXOK1: Transmission OK MailBox1 Flag
  - CAN\_FLAG\_TXOK2: Transmission OK MailBox2 Flag
  - CAN\_FLAG\_TME0: Transmit mailbox 0 empty Flag
  - CAN\_FLAG\_TME1: Transmit mailbox 1 empty Flag
  - CAN\_FLAG\_TME2: Transmit mailbox 2 empty Flag
  - CAN\_FLAG\_FMP0: FIFO 0 Message Pending Flag
  - CAN\_FLAG\_FF0: FIFO 0 Full Flag
  - CAN\_FLAG\_FOV0: FIFO 0 Overrun Flag
  - CAN\_FLAG\_FMP1: FIFO 1 Message Pending Flag
  - CAN\_FLAG\_FF1: FIFO 1 Full Flag
  - CAN\_FLAG\_FOV1: FIFO 1 Overrun Flag
  - CAN\_FLAG\_WKU: Wake up Flag
  - CAN\_FLAG\_SLAK: Sleep acknowledge Flag
  - CAN\_FLAG\_SLAKI: Sleep acknowledge Flag
  - CAN\_FLAG\_EWG: Error Warning Flag
  - CAN\_FLAG\_EPV: Error Passive Flag
  - CAN\_FLAG\_BOF: Bus-Off Flag

**Return value:**

- The new state of [\\_\\_FLAG\\_\\_](#) (TRUE or FALSE).

[\\_\\_HAL\\_CAN\\_CLEAR\\_FLAG](#)**Description:**

- Clear the specified CAN pending flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the CAN Handle.
- \_\_FLAG\_\_: specifies the flag to check.  
This parameter can be one of the following values:
  - CAN\_TSR\_RQCP0: Request MailBox0 Flag
  - CAN\_TSR\_RQCP1: Request MailBox1 Flag
  - CAN\_TSR\_RQCP2: Request MailBox2 Flag
  - CAN\_FLAG\_TXOK0: Transmission OK MailBox0 Flag
  - CAN\_FLAG\_TXOK1: Transmission OK MailBox1 Flag
  - CAN\_FLAG\_TXOK2: Transmission OK MailBox2 Flag
  - CAN\_FLAG\_TME0: Transmit mailbox 0 empty Flag
  - CAN\_FLAG\_TME1: Transmit mailbox 1 empty Flag
  - CAN\_FLAG\_TME2: Transmit mailbox 2 empty Flag
  - CAN\_FLAG\_FMP0: FIFO 0 Message Pending Flag
  - CAN\_FLAG\_FF0: FIFO 0 Full Flag
  - CAN\_FLAG\_FOV0: FIFO 0 Overrun Flag
  - CAN\_FLAG\_FMP1: FIFO 1 Message Pending Flag
  - CAN\_FLAG\_FF1: FIFO 1 Full Flag
  - CAN\_FLAG\_FOV1: FIFO 1 Overrun Flag
  - CAN\_FLAG\_WKU: Wake up Flag
  - CAN\_FLAG\_SLAKI: Sleep acknowledge Flag
  - CAN\_FLAG\_EWG: Error Warning Flag
  - CAN\_FLAG\_EPV: Error Passive Flag
  - CAN\_FLAG\_BOF: Bus-Off Flag

**Return value:**

- The new state of \_\_FLAG\_\_ (TRUE or FALSE).

[\\_\\_HAL\\_CAN\\_GET\\_IT\\_SOURCE](#)**Description:**

- Check if the specified CAN interrupt source is enabled or disabled.

**Parameters:**

- \_\_HANDLE\_\_: specifies the CAN Handle.
- \_\_INTERRUPT\_\_: specifies the CAN interrupt source to check. This parameter

can be one of the following values:

- CAN\_IT\_TME: Transmit mailbox empty interrupt enable
- CAN\_IT\_FMP0: FIFO0 message pending interrupt enable
- CAN\_IT\_FMP1: FIFO1 message pending interrupt enable

**Return value:**

- The: new state of \_\_IT\_\_ (TRUE or FALSE).

[\\_\\_HAL\\_CAN\\_TRANSMIT\\_STATUS](#)

**Description:**

- Check the transmission status of a CAN Frame.

**Parameters:**

- \_\_HANDLE\_\_: CAN handle.
- \_\_TRANSMITMAILBOX\_\_: the number of the mailbox that is used for transmission.

**Return value:**

- The: new status of transmission (TRUE or FALSE).

[\\_\\_HAL\\_CAN\\_FIFO\\_RELEASE](#)

**Description:**

- Release the specified receive FIFO.

**Parameters:**

- \_\_HANDLE\_\_: CAN handle.
- \_\_FIFONUMBER\_\_: Receive FIFO number, CAN\_FIFO0 or CAN\_FIFO1.

**Return value:**

- None

[\\_\\_HAL\\_CAN\\_CANCEL\\_TRANSMIT](#)

**Description:**

- Cancel a transmit request.

**Parameters:**

- \_\_HANDLE\_\_: specifies the CAN Handle.
- \_\_TRANSMITMAILBOX\_\_: the number of the mailbox that is used for transmission.

**Return value:**

- None

[\\_\\_HAL\\_CAN\\_DBG\\_FREEZE](#)

**Description:**

- Enable or disables the DBG Freeze for CAN.

**Parameters:**

- \_\_HANDLE\_\_: specifies the CAN Handle.
- \_\_NEWSTATE\_\_: new state of the CAN

peripheral. This parameter can be:  
ENABLE (CAN reception/transmission is frozen during debug. Reception FIFOs can still be accessed/controlled normally) or  
DISABLE (CAN is working during debug).

**Return value:**

- None

**CAN Filter FIFO**

|                  |                                       |
|------------------|---------------------------------------|
| CAN_FILTER_FIFO0 | Filter FIFO 0 assignment for filter x |
| CAN_FILTER_FIFO1 | Filter FIFO 1 assignment for filter x |

**CAN Filter Mode**

|                       |                      |
|-----------------------|----------------------|
| CAN_FILTERMODE_IDMASK | Identifier mask mode |
| CAN_FILTERMODE_IDLIST | Identifier list mode |

**CAN Filter Scale**

|                       |                    |
|-----------------------|--------------------|
| CAN_FILTERSCALE_16BIT | Two 16-bit filters |
| CAN_FILTERSCALE_32BIT | One 32-bit filter  |

**CAN Flags**

|                |                                 |
|----------------|---------------------------------|
| CAN_FLAG_RQCP0 | Request MailBox0 flag           |
| CAN_FLAG_RQCP1 | Request MailBox1 flag           |
| CAN_FLAG_RQCP2 | Request MailBox2 flag           |
| CAN_FLAG_TXOK0 | Transmission OK MailBox0 flag   |
| CAN_FLAG_TXOK1 | Transmission OK MailBox1 flag   |
| CAN_FLAG_TXOK2 | Transmission OK MailBox2 flag   |
| CAN_FLAG_TME0  | Transmit mailbox 0 empty flag   |
| CAN_FLAG_TME1  | Transmit mailbox 0 empty flag   |
| CAN_FLAG_TME2  | Transmit mailbox 0 empty flag   |
| CAN_FLAG_FF0   | FIFO 0 Full flag                |
| CAN_FLAG_FOV0  | FIFO 0 Overrun flag             |
| CAN_FLAG_FF1   | FIFO 1 Full flag                |
| CAN_FLAG_FOV1  | FIFO 1 Overrun flag             |
| CAN_FLAG_INAK  | Initialization acknowledge flag |
| CAN_FLAG_SLAK  | Sleep acknowledge flag          |
| CAN_FLAG_ERRI  | Error flag                      |
| CAN_FLAG_WKU   | Wake up flag                    |
| CAN_FLAG_SLAKI | Sleep acknowledge flag          |
| CAN_FLAG_EWG   | Error warning flag              |
| CAN_FLAG_EPV   | Error passive flag              |
| CAN_FLAG_BOF   | Bus-Off flag                    |

**CAN Identifier Type**

|            |             |
|------------|-------------|
| CAN_ID_STD | Standard Id |
| CAN_ID_EXT | Extended Id |

**CAN InitStatus**

|                        |                           |
|------------------------|---------------------------|
| CAN_INITSTATUS_FAILED  | CAN initialization failed |
| CAN_INITSTATUS_SUCCESS | CAN initialization OK     |

**CAN Interrupts**

|             |                                  |
|-------------|----------------------------------|
| CAN_IT_TME  | Transmit mailbox empty interrupt |
| CAN_IT_FMP0 | FIFO 0 message pending interrupt |
| CAN_IT_FF0  | FIFO 0 full interrupt            |
| CAN_IT_FOV0 | FIFO 0 overrun interrupt         |
| CAN_IT_FMP1 | FIFO 1 message pending interrupt |
| CAN_IT_FF1  | FIFO 1 full interrupt            |
| CAN_IT_FOV1 | FIFO 1 overrun interrupt         |
| CAN_IT_WKU  | Wake-up interrupt                |
| CAN_IT_SLK  | Sleep acknowledge interrupt      |
| CAN_IT_EWG  | Error warning interrupt          |
| CAN_IT_EPV  | Error passive interrupt          |
| CAN_IT_BOF  | Bus-off interrupt                |
| CAN_IT_LEC  | Last error code interrupt        |
| CAN_IT_ERR  | Error Interrupt                  |

**CAN Mailboxes**

|                 |
|-----------------|
| CAN_TXMAILBOX_0 |
| CAN_TXMAILBOX_1 |
| CAN_TXMAILBOX_2 |

**CAN Operating Mode**

|                          |                                    |
|--------------------------|------------------------------------|
| CAN_MODE_NORMAL          | Normal mode                        |
| CAN_MODE_LOOPBACK        | Loopback mode                      |
| CAN_MODE_SILENT          | Silent mode                        |
| CAN_MODE_SILENT_LOOPBACK | Loopback combined with silent mode |

**CAN Receive FIFO Number**

|           |                            |
|-----------|----------------------------|
| CAN_FIFO0 | CAN FIFO 0 used to receive |
| CAN_FIFO1 | CAN FIFO 1 used to receive |

**CAN Remote Transmission Request**

|                |              |
|----------------|--------------|
| CAN_RTR_DATA   | Data frame   |
| CAN_RTR_REMOTE | Remote frame |

**CAN Synchronization Jump Width**

|             |                |
|-------------|----------------|
| CAN_SJW_1TQ | 1 time quantum |
| CAN_SJW_2TQ | 2 time quantum |
| CAN_SJW_3TQ | 3 time quantum |
| CAN_SJW_4TQ | 4 time quantum |

**CAN Time Quantum in Bit Segment 1**

|              |                 |
|--------------|-----------------|
| CAN_BS1_1TQ  | 1 time quantum  |
| CAN_BS1_2TQ  | 2 time quantum  |
| CAN_BS1_3TQ  | 3 time quantum  |
| CAN_BS1_4TQ  | 4 time quantum  |
| CAN_BS1_5TQ  | 5 time quantum  |
| CAN_BS1_6TQ  | 6 time quantum  |
| CAN_BS1_7TQ  | 7 time quantum  |
| CAN_BS1_8TQ  | 8 time quantum  |
| CAN_BS1_9TQ  | 9 time quantum  |
| CAN_BS1_10TQ | 10 time quantum |
| CAN_BS1_11TQ | 11 time quantum |
| CAN_BS1_12TQ | 12 time quantum |
| CAN_BS1_13TQ | 13 time quantum |
| CAN_BS1_14TQ | 14 time quantum |
| CAN_BS1_15TQ | 15 time quantum |
| CAN_BS1_16TQ | 16 time quantum |

**CAN Time Quantum in Bit Segment 2**

|             |                |
|-------------|----------------|
| CAN_BS2_1TQ | 1 time quantum |
| CAN_BS2_2TQ | 2 time quantum |
| CAN_BS2_3TQ | 3 time quantum |
| CAN_BS2_4TQ | 4 time quantum |
| CAN_BS2_5TQ | 5 time quantum |
| CAN_BS2_6TQ | 6 time quantum |
| CAN_BS2_7TQ | 7 time quantum |
| CAN_BS2_8TQ | 8 time quantum |

## 9 HAL CEC Generic Driver

### 9.1 CEC Firmware driver registers structures

#### 9.1.1 CEC\_InitTypeDef

##### Data Fields

- *uint32\_t SignalFreeTime*
- *uint32\_t Tolerance*
- *uint32\_t BRERxStop*
- *uint32\_t BREErrorBitGen*
- *uint32\_t LBPEErrorBitGen*
- *uint32\_t BroadcastMsgNoErrorBitGen*
- *uint32\_t SignalFreeTimeOption*
- *uint32\_t ListenMode*
- *uint16\_t OwnAddress*
- *uint8\_t \* RxBuffer*

##### Field Documentation

- ***uint32\_t CEC\_InitTypeDef::SignalFreeTime***  
Set SFT field, specifies the Signal Free Time. It can be one of **CEC\_Signal\_Free\_Time** and belongs to the set {0U,...,7} where 0x0 is the default configuration else means 0.5U + (SignalFreeTime - 1U) nominal data bit periods
- ***uint32\_t CEC\_InitTypeDef::Tolerance***  
Set RXTOL bit, specifies the tolerance accepted on the received waveforms, it can be a value of **CEC\_Tolerance** : it is either CEC\_STANDARD\_TOLERANCE or CEC\_EXTENDED\_TOLERANCE
- ***uint32\_t CEC\_InitTypeDef::BRERxStop***  
Set BRESTRP bit **CEC\_BRERxStop** : specifies whether or not a Bit Rising Error stops the reception. CEC\_NO\_RX\_STOP\_ON\_BRE: reception is not stopped. CEC\_RX\_STOP\_ON\_BRE: reception is stopped.
- ***uint32\_t CEC\_InitTypeDef::BREErrorBitGen***  
Set BREGEN bit **CEC\_BREErrorBitGen** : specifies whether or not an Error-Bit is generated on the CEC line upon Bit Rising Error detection.  
CEC\_BRE\_ERRORBIT\_NO\_GENERATION: no error-bit generation.  
CEC\_BRE\_ERRORBIT\_GENERATION: error-bit generation if BRESTRP is set.
- ***uint32\_t CEC\_InitTypeDef::LBPEErrorBitGen***  
Set LBPEGEN bit **CEC\_LBPEErrorBitGen** : specifies whether or not an Error-Bit is generated on the CEC line upon Long Bit Period Error detection.  
CEC\_LBPE\_ERRORBIT\_NO\_GENERATION: no error-bit generation.  
CEC\_LBPE\_ERRORBIT\_GENERATION: error-bit generation.
- ***uint32\_t CEC\_InitTypeDef::BroadcastMsgNoErrorBitGen***  
Set BRDNOGEN bit **CEC\_BroadcastMsgErrorBitGen** : allows to avoid an Error-Bit generation on the CEC line upon an error detected on a broadcast message. It supersedes BREGEN and LBPEGEN bits for a broadcast message error handling. It can take two values:1U) CEC\_BROADCASTERROR\_ERRORBIT\_GENERATION. a) BRE detection: error-bit generation on the CEC line if BRESTRP=CEC\_RX\_STOP\_ON\_BRE and BREGEN=CEC\_BRE\_ERRORBIT\_NO\_GENERATION. b) LBPE detection: error-bit generation on the CEC line if LBPGEN=CEC\_LBPE\_ERRORBIT\_NO\_GENERATION.2U)

CEC\_BROADCASTERROR\_NO\_ERRORBIT\_GENERATION. no error-bit generation in case neither a) nor b) are satisfied. Additionally, there is no error-bit generation in case of Short Bit Period Error detection in a broadcast message while LSTN bit is set.

- ***uint32\_t CEC\_InitTypeDef::SignalFreeTimeOption***  
Set SFTOP bit ***CEC\_SFT\_Option*** : specifies when SFT timer starts.  
CEC\_SFT\_START\_ON\_TXSOM SFT: timer starts when TXSOM is set by software.  
CEC\_SFT\_START\_ON\_RX\_END: SFT timer starts automatically at the end of message transmission/reception.
- ***uint32\_t CEC\_InitTypeDef::ListenMode***  
Set LSTN bit ***CEC\_Listening\_Mode*** : specifies device listening mode. It can take two values:  
CEC\_REDUCED\_LISTENING\_MODE: CEC peripheral receives only message addressed to its own address (OAR). Messages addressed to different destination are ignored. Broadcast messages are always received.  
CEC\_FULL\_LISTENING\_MODE: CEC peripheral receives messages addressed to its own address (OAR) with positive acknowledge. Messages addressed to different destination are received, but without interfering with the CEC bus: no acknowledge sent.
- ***uint16\_t CEC\_InitTypeDef::OwnAddress***  
Own addresses configuration This parameter can be a value of  
***CEC\_OWN\_ADDRESS***
- ***uint8\_t\* CEC\_InitTypeDef::RxBuffer***  
CEC Rx buffer pointeur

## 9.1.2 CEC\_HandleTypeDef

### Data Fields

- ***CEC\_TypeDef \* Instance***
- ***CEC\_InitTypeDef Init***
- ***uint8\_t \* pTxBuffPtr***
- ***uint16\_t TxXferCount***
- ***uint16\_t RxXferSize***
- ***HAL\_LockTypeDef Lock***
- ***HAL\_CEC\_StateTypeDef gState***
- ***HAL\_CEC\_StateTypeDef RxState***
- ***uint32\_t ErrorCode***

### Field Documentation

- ***CEC\_TypeDef\* CEC\_HandleTypeDef::Instance***  
CEC registers base address
- ***CEC\_InitTypeDef CEC\_HandleTypeDef::Init***  
CEC communication parameters
- ***uint8\_t\* CEC\_HandleTypeDef::pTxBuffPtr***  
Pointer to CEC Tx transfer Buffer
- ***uint16\_t CEC\_HandleTypeDef::TxXferCount***  
CEC Tx Transfer Counter
- ***uint16\_t CEC\_HandleTypeDef::RxXferSize***  
CEC Rx Transfer size, 0: header received only
- ***HAL\_LockTypeDef CEC\_HandleTypeDef::Lock***  
Locking object
- ***HAL\_CEC\_StateTypeDef CEC\_HandleTypeDef::gState***  
CEC state information related to global Handle management and also related to Tx operations. This parameter can be a value of ***HAL\_CEC\_StateTypeDef***
- ***HAL\_CEC\_StateTypeDef CEC\_HandleTypeDef::RxState***  
CEC state information related to Rx operations. This parameter can be a value of ***HAL\_CEC\_StateTypeDef***

- ***uint32\_t CEC\_HandleTypeDef::ErrorCode***  
For errors handling purposes, copy of ISR register in case error is reported

## 9.2 CEC Firmware driver API description

### 9.2.1 How to use this driver



The specific CEC interrupts (Transmission complete interrupt, RXNE interrupt and Error Interrupts) will be managed using the macros \_\_HAL\_CEC\_ENABLE\_IT() and \_\_HAL\_CEC\_DISABLE\_IT() inside the transmit and receive process. (#) Program the Signal Free Time (SFT) and SFT option, Tolerance, reception stop in in case of Bit Rising Error, Error-Bit generation conditions, device logical address and Listen mode in the hcec Init structure. (#) Initialize the CEC registers by calling the HAL\_CEC\_Init() API.



This API (HAL\_CEC\_Init()) configures also the low level Hardware GPIO, CLOCK, CORTEX...etc by calling the customized HAL\_CEC\_MspInit() API.

### 9.2.2 Initialization and Configuration functions

This subsection provides a set of functions allowing to initialize the CEC

- The following parameters need to be configured:
  - SignalFreeTime
  - Tolerance
  - BRERxStop (RX stopped or not upon Bit Rising Error)
  - BREErrorBitGen (Error-Bit generation in case of Bit Rising Error)
  - LBPEErrorBitGen (Error-Bit generation in case of Long Bit Period Error)
  - BroadcastMsgNoErrorBitGen (Error-bit generation in case of broadcast message error)
  - SignalFreeTimeOption (SFT Timer start definition)
  - OwnAddress (CEC device address)
  - ListenMode

This section contains the following APIs:

- [\*\*HAL\\_CEC\\_Init\(\)\*\*](#)
- [\*\*HAL\\_CEC\\_DelInit\(\)\*\*](#)
- [\*\*HAL\\_CEC\\_SetDeviceAddress\(\)\*\*](#)
- [\*\*HAL\\_CEC\\_MspInit\(\)\*\*](#)
- [\*\*HAL\\_CEC\\_MspDelInit\(\)\*\*](#)

### 9.2.3 IO operation functions

This section contains the following APIs:

- [\*\*HAL\\_CEC\\_Transmit\\_IT\(\)\*\*](#)
- [\*\*HAL\\_CEC\\_GetLastReceivedFrameSize\(\)\*\*](#)
- [\*\*HAL\\_CEC\\_ChangeRxBuffer\(\)\*\*](#)
- [\*\*HAL\\_CEC\\_IRQHandler\(\)\*\*](#)
- [\*\*HAL\\_CEC\\_TxCpltCallback\(\)\*\*](#)
- [\*\*HAL\\_CEC\\_RxCpltCallback\(\)\*\*](#)

- [\*HAL\\_CEC\\_ErrorCallback\(\)\*](#)

## 9.2.4 Peripheral Control function

This subsection provides a set of functions allowing to control the CEC.

- HAL\_CEC\_GetState() API can be helpful to check in run-time the state of the CEC peripheral.
- HAL\_CEC\_GetError() API can be helpful to check in run-time the error of the CEC peripheral.

This section contains the following APIs:

- [\*HAL\\_CEC\\_GetState\(\)\*](#)
- [\*HAL\\_CEC\\_GetError\(\)\*](#)

## 9.2.5 Detailed description of functions

### **HAL\_CEC\_Init**

|                      |                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CEC_Init (CEC_HandleTypeDef * hcec)</b>                                                          |
| Function description | Initializes the CEC mode according to the specified parameters in the CEC_InitTypeDef and creates the associated handle . |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcec:</b> CEC handle</li> </ul>                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                    |

### **HAL\_CEC\_DelInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CEC_DelInit (CEC_HandleTypeDef * hcec)</b>         |
| Function description | Delinitializes the CEC peripheral.                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcec:</b> CEC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>      |

### **HAL\_CEC\_SetDeviceAddress**

|                      |                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CEC_SetDeviceAddress (CEC_HandleTypeDef * hcec, uint16_t CEC_OwnAddress)</b>                              |
| Function description | Initializes the Own Address of the CEC device.                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcec:</b> CEC handle</li> <li>• <b>CEC_OwnAddress:</b> The CEC own address.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                             |

### **HAL\_CEC\_MspInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_CEC_MspInit (CEC_HandleTypeDef * hcec)</b>                      |
| Function description | CEC MSP Init.                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcec:</b> CEC handle</li> </ul> |

|               |                                                               |
|---------------|---------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"><li>• <b>None</b></li></ul> |
|---------------|---------------------------------------------------------------|

### HAL\_CEC\_MspDelInit

Function name      **void HAL\_CEC\_MspDelInit (CEC\_HandleTypeDef \* hcec)**

Function description      CEC MSP DelInit.

Parameters      

- **hcec:** CEC handle

Return values      

- **None**

### HAL\_CEC\_Transmit\_IT

Function name      **HAL\_StatusTypeDef HAL\_CEC\_Transmit\_IT  
(CEC\_HandleTypeDef \* hcec, uint8\_t InitiatorAddress, uint8\_t  
DestinationAddress, uint8\_t \* pData, uint32\_t Size)**

Function description      Send data in interrupt mode.

Parameters      

- **hcec:** CEC handle
- **InitiatorAddress:** Initiator address
- **DestinationAddress:** destination logical address
- **pData:** pointer to input byte data buffer
- **Size:** amount of data to be sent in bytes (without counting the header). 0 means only the header is sent (ping operation). Maximum TX size is 15 bytes (1 opcode and up to 14 operands).

Return values      

- **HAL:** status

### HAL\_CEC\_GetLastReceivedFrameSize

Function name      **uint32\_t HAL\_CEC\_GetLastReceivedFrameSize  
(CEC\_HandleTypeDef \* hcec)**

Function description      Get size of the received frame.

Parameters      

- **hcec:** CEC handle

Return values      

- **Frame:** size

### HAL\_CEC\_ChangeRxBuffer

Function name      **void HAL\_CEC\_ChangeRxBuffer (CEC\_HandleTypeDef \* hcec,  
uint8\_t \* Rxbuffer)**

Function description      Change Rx Buffer.

Parameters      

- **hcec:** CEC handle
- **Rxbuffer:** Rx Buffer

Return values      

- **Frame:** size

Notes      

- This function can be called only inside the HAL\_CEC\_RxCpltCallback()

**HAL\_CEC\_IRQHandler**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_CEC_IRQHandler (CEC_HandleTypeDef * hcec)</b>                   |
| Function description | This function handles CEC interrupt requests.                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcec:</b> CEC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_CEC\_TxCpltCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_CEC_TxCpltCallback (CEC_HandleTypeDef * hcec)</b>               |
| Function description | Tx Transfer completed callback.                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcec:</b> CEC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_CEC\_RxCpltCallback**

|                      |                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_CEC_RxCpltCallback (CEC_HandleTypeDef * hcec, uint32_t RxFrameSize)</b>                                      |
| Function description | Rx Transfer completed callback.                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcec:</b> CEC handle</li> <li>• <b>RxFrameSize:</b> Size of frame</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                          |

**HAL\_CEC\_ErrorCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_CEC_ErrorCallback (CEC_HandleTypeDef * hcec)</b>                |
| Function description | CEC error callbacks.                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcec:</b> CEC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_CEC\_GetState**

|                      |                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_CEC_StateTypeDef HAL_CEC_GetState (CEC_HandleTypeDef * hcec)</b>                                                                                                             |
| Function description | return the CEC state                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcec:</b> pointer to a CEC_HandleTypeDef structure that contains the configuration information for the specified CEC module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                               |

**HAL\_CEC\_GetError**

|                      |                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_CEC_GetError (CEC_HandleTypeDef * hcec)</b>                                                    |
| Function description | Return the CEC error code.                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcec:</b> pointer to a CEC_HandleTypeDef structure that</li> </ul> |

contains the configuration information for the specified CEC.

Return values

- **CEC:** Error Code

## 9.3 CEC Firmware driver defines

### 9.3.1 CEC

**CEC all RX or TX errors flags**

`CEC_ISR_ALL_ERROR`

**CEC Error Bit Generation if Bit Rise Error reported**

`CEC_BRE_ERRORBIT_NO_GENERATION`

`CEC_BRE_ERRORBIT_GENERATION`

**CEC Reception Stop on Error**

`CEC_NO_RX_STOP_ON_BRE`

`CEC_RX_STOP_ON_BRE`

**CEC Error Bit Generation on Broadcast message**

`CEC_BROADCASTERROR_ERRORBIT_GENERATION`

`CEC_BROADCASTERROR_NO_ERRORBIT_GENERATION`

**CEC Error Code**

`HAL_CEC_ERROR_NONE` no error

`HAL_CEC_ERROR_RXOVR` CEC Rx-Overrun

`HAL_CEC_ERROR_BRE` CEC Rx Bit Rising Error

`HAL_CEC_ERROR_SBPE` CEC Rx Short Bit period Error

`HAL_CEC_ERROR_LBPE` CEC Rx Long Bit period Error

`HAL_CEC_ERROR_RXACKE` CEC Rx Missing Acknowledge

`HAL_CEC_ERROR_ARBLST` CEC Arbitration Lost

`HAL_CEC_ERROR_TXUDR` CEC Tx-Buffer Underrun

`HAL_CEC_ERROR_TXERR` CEC Tx-Error

`HAL_CEC_ERROR_TXACKE` CEC Tx Missing Acknowledge

**CEC Exported Macros**

`_HAL_CEC_RESET_HANDLE_STATE`

**Description:**

- Reset CEC handle gstate & RxState.

**Parameters:**

- `_HANDLE_`: CEC handle.

**Return value:**

- None

**Description:**

- Checks whether or not the

`_HAL_CEC_GET_FLAG`

specified CEC interrupt flag is set.

**Parameters:**

- \_\_HANDLE\_\_: specifies the CEC Handle.
- \_\_FLAG\_\_: specifies the flag to check.
  - CEC\_FLAG\_TXACKE: Tx Missing acknowledge Error
  - CEC\_FLAG\_TXERR: Tx Error.
  - CEC\_FLAG\_RXUDR: Rx-Buffer Underrun.
  - CEC\_FLAG\_TXEND: End of transmission (successful transmission of the last byte).
  - CEC\_FLAG\_RXBR: Rx-Byte Request.
  - CEC\_FLAG\_ARBLST: Arbitration Lost
  - CEC\_FLAG\_RXACKE: Rx-Missing Acknowledge
  - CEC\_FLAG\_LBPE: Rx Long period Error
  - CEC\_FLAG\_SBPE: Rx Short period Error
  - CEC\_FLAG\_BRE: Rx Bit Rising Error
  - CEC\_FLAG\_RXOVR: Rx Overrun.
  - CEC\_FLAG\_RXEND: End Of Reception.
  - CEC\_FLAG\_RXBR: Rx-Byte Received.

**Return value:**

- ITStatus

[\\_\\_HAL\\_CEC\\_CLEAR\\_FLAG](#)

**Description:**

- Clears the interrupt or status flag when raised (write at 1U)

**Parameters:**

- \_\_HANDLE\_\_: specifies the CEC Handle.
- \_\_FLAG\_\_: specifies the interrupt/status flag to clear. This parameter can be one of the following values:
  - CEC\_FLAG\_TXACKE: Tx Missing acknowledge Error
  - CEC\_FLAG\_TXERR: Tx

- Error.
- CEC\_FLAG\_TXUDR: Tx-Buffer Underrun.
- CEC\_FLAG\_TXEND: End of transmission (successful transmission of the last byte).
- CEC\_FLAG\_RXBR: Rx-Byte Request.
- CEC\_FLAG\_ARBLST: Arbitration Lost
- CEC\_FLAG\_RXACKE: Rx-Missing Acknowledge
- CEC\_FLAG\_LBPE: Rx Long period Error
- CEC\_FLAG\_SBPE: Rx Short period Error
- CEC\_FLAG\_BRE: Rx Bit Rising Error
- CEC\_FLAG\_RXOVR: Rx Overrun.
- CEC\_FLAG\_RXEND: End Of Reception.
- CEC\_FLAG\_RXBR: Rx-Byte Received.

**Return value:**

- none

**\_HAL\_CEC\_ENABLE\_IT****Description:**

- Enables the specified CEC interrupt.

**Parameters:**

- \_\_HANDLE\_\_: specifies the CEC Handle.
- \_\_INTERRUPT\_\_: specifies the CEC interrupt to enable. This parameter can be one of the following values:
  - CEC\_IT\_RXACKE: Rx-Missing acknowledge Error IT Enable
  - CEC\_IT\_TXERR: Tx Error IT Enable
  - CEC\_IT\_TXUDR: Tx-Buffer Underrun IT Enable
  - CEC\_IT\_TXEND: End of transmission IT Enable
  - CEC\_IT\_RXBR: Rx-Byte Request IT Enable
  - CEC\_IT\_ARBLST: Arbitration Lost IT Enable
  - CEC\_IT\_RXACKE: Rx-Missing acknowledge Error IT Enable

- Missing Acknowledge IT Enable
- CEC\_IT\_LBPE: Rx Long period Error IT Enable
- CEC\_IT\_SBPE: Rx Short period Error IT Enable
- CEC\_IT\_BRE: Rx Bit Rising Error IT Enable
- CEC\_IT\_RXOVR: Rx Overrun IT Enable
- CEC\_IT\_RXEND: End Of Reception IT Enable
- CEC\_IT\_RXBR: Rx-Byte Received IT Enable

**Return value:**

- none

**\_HAL\_CEC\_DISABLE\_IT****Description:**

- Disables the specified CEC interrupt.

**Parameters:**

- \_HANDLE\_: specifies the CEC Handle.
- \_INTERRUPT\_: specifies the CEC interrupt to disable. This parameter can be one of the following values:
  - CEC\_IT\_TXACKE: Tx Missing acknowledge Error IT Enable
  - CEC\_IT\_TXERR: Tx Error IT Enable
  - CEC\_IT\_TXUDR: Tx-Buffer Underrun IT Enable
  - CEC\_IT\_TXEND: End of transmission IT Enable
  - CEC\_IT\_TXBR: Tx-Byte Request IT Enable
  - CEC\_IT\_ARBLST: Arbitration Lost IT Enable
  - CEC\_IT\_RXACKE: Rx-Missing Acknowledge IT Enable
  - CEC\_IT\_LBPE: Rx Long period Error IT Enable
  - CEC\_IT\_SBPE: Rx Short period Error IT Enable
  - CEC\_IT\_BRE: Rx Bit Rising Error IT Enable
  - CEC\_IT\_RXOVR: Rx Overrun IT Enable
  - CEC\_IT\_RXEND: End Of

- Reception IT Enable
- CEC\_IT\_RXBR: Rx-Byte Received IT Enable

**Return value:**

- none

**\_HAL\_CEC\_GET\_IT\_SOURCE****Description:**

- Checks whether or not the specified CEC interrupt is enabled.

**Parameters:**

- \_\_HANDLE\_\_: specifies the CEC Handle.
- \_\_INTERRUPT\_\_: specifies the CEC interrupt to check. This parameter can be one of the following values:
  - CEC\_IT\_TXACKE: Tx Missing acknowledge Error IT Enable
  - CEC\_IT\_TXERR: Tx Error IT Enable
  - CEC\_IT\_TXUDR: Tx-Buffer Underrun IT Enable
  - CEC\_IT\_TXEND: End of transmission IT Enable
  - CEC\_IT\_TXBR: Tx-Byte Request IT Enable
  - CEC\_IT\_ARBLST: Arbitration Lost IT Enable
  - CEC\_IT\_RXACKE: Rx-Missing Acknowledge IT Enable
  - CEC\_IT\_LBPE: Rx Long period Error IT Enable
  - CEC\_IT\_SBPE: Rx Short period Error IT Enable
  - CEC\_IT\_BRE: Rx Bit Rising Error IT Enable
  - CEC\_IT\_RXOVR: Rx Overrun IT Enable
  - CEC\_IT\_RXEND: End Of Reception IT Enable
  - CEC\_IT\_RXBR: Rx-Byte Received IT Enable

**Return value:**

- FlagStatus

**\_HAL\_CEC\_ENABLE****Description:**

- Enables the CEC device.

**Parameters:**

- `__HANDLE__`: specifies the CEC Handle.

**Return value:**

- none

`__HAL_CEC_DISABLE`

**Description:**

- Disables the CEC device.

**Parameters:**

- `__HANDLE__`: specifies the CEC Handle.

**Return value:**

- none

`__HAL_CEC_FIRST_BYTE_TX_SET`

**Description:**

- Set Transmission Start flag.

**Parameters:**

- `__HANDLE__`: specifies the CEC Handle.

**Return value:**

- none

`__HAL_CEC_LAST_BYTE_TX_SET`

**Description:**

- Set Transmission End flag.

**Parameters:**

- `__HANDLE__`: specifies the CEC Handle.

**Return value:**

- None If the CEC message consists of only one byte, TXEOM must be set before of TXSOM.

`__HAL_CEC_GET_TRANSMISSION_START_FLAG`

**Description:**

- Get Transmission Start flag.

**Parameters:**

- `__HANDLE__`: specifies the CEC Handle.

**Return value:**

- FlagStatus

`__HAL_CEC_GET_TRANSMISSION_END_FLAG`

**Description:**

- Get Transmission End flag.

**Parameters:**

- `__HANDLE__`: specifies the CEC Handle.

**Return value:**

- `FlagStatus`

**Description:**

- Clear OAR register.

**Parameters:**

- `__HANDLE__`: specifies the CEC Handle.

**Return value:**

- `none`

**Description:**

- Set OAR register (without resetting previously set address in case of multi-address mode)  
To reset OAR,

**Parameters:**

- `__HANDLE__`: specifies the CEC Handle.
- `__ADDRESS__`: Own Address value (CEC logical address is identified by bit position)

**Return value:**

- `none`

***CEC Flags definition***

`CEC_FLAG_TXACKE`  
`CEC_FLAG_TXERR`  
`CEC_FLAG_TXUDR`  
`CEC_FLAG_TXEND`  
`CEC_FLAG_TXBR`  
`CEC_FLAG_ARBLST`  
`CEC_FLAG_RXACKE`  
`CEC_FLAG_LBPE`  
`CEC_FLAG_SBPE`  
`CEC_FLAG_BRE`  
`CEC_FLAG_RXOVR`  
`CEC_FLAG_RXEND`  
`CEC_FLAG_RXBR`

***CEC all RX errors interrupts enabling flag***

CEC\_IER\_RX\_ALL\_ERR

***CEC all TX errors interrupts enabling flag***

CEC\_IER\_TX\_ALL\_ERR

***CEC Initiator logical address position in message header***

CEC\_INITIATOR\_LSB\_POS

***CEC Interrupts definition***

CEC\_IT\_TXACKE

CEC\_IT\_TXERR

CEC\_IT\_TXUDR

CEC\_IT\_TXEND

CEC\_IT\_TXBR

CEC\_IT\_ARBLST

CEC\_IT\_RXACKE

CEC\_IT\_LBPE

CEC\_IT\_SBPE

CEC\_IT\_BRE

CEC\_IT\_RXOVR

CEC\_IT\_RXEND

CEC\_IT\_RXBR

***CEC Error Bit Generation if Long Bit Period Error reported***

CEC\_LBPE\_ERRORBIT\_NO\_GENERATION

CEC\_LBPE\_ERRORBIT\_GENERATION

***CEC Listening mode option***

CEC\_REDUCED\_LISTENING\_MODE

CEC\_FULL\_LISTENING\_MODE

***CEC Device Own Address position in CEC CFGR register***

CEC\_CFGR\_OAR\_LSB\_POS

***CEC Own Address***

CEC\_OWN\_ADDRESS\_NONE

CEC\_OWN\_ADDRESS\_0

CEC\_OWN\_ADDRESS\_1

CEC\_OWN\_ADDRESS\_2

CEC\_OWN\_ADDRESS\_3

CEC\_OWN\_ADDRESS\_4

CEC\_OWN\_ADDRESS\_5

CEC\_OWN\_ADDRESS\_6  
CEC\_OWN\_ADDRESS\_7  
CEC\_OWN\_ADDRESS\_8  
CEC\_OWN\_ADDRESS\_9  
CEC\_OWN\_ADDRESS\_10  
CEC\_OWN\_ADDRESS\_11  
CEC\_OWN\_ADDRESS\_12  
CEC\_OWN\_ADDRESS\_13  
CEC\_OWN\_ADDRESS\_14

***CEC Signal Free Time start option***

CEC\_SFT\_START\_ON\_TXSOM  
CEC\_SFT\_START\_ON\_TX\_RX\_END

***CEC Signal Free Time setting parameter***

CEC\_DEFAULT\_SFT  
CEC\_0\_5\_BITPERIOD\_SFT  
CEC\_1\_5\_BITPERIOD\_SFT  
CEC\_2\_5\_BITPERIOD\_SFT  
CEC\_3\_5\_BITPERIOD\_SFT  
CEC\_4\_5\_BITPERIOD\_SFT  
CEC\_5\_5\_BITPERIOD\_SFT  
CEC\_6\_5\_BITPERIOD\_SFT

***CEC Receiver Tolerance***

CEC\_STANDARD\_TOLERANCE  
CEC\_EXTENDED\_TOLERANCE

## 10 HAL COMP Generic Driver

### 10.1 COMP Firmware driver registers structures

#### 10.1.1 COMP\_InitTypeDef

##### Data Fields

- *uint32\_t InvertingInput*
- *uint32\_t NonInvertingInput*
- *uint32\_t Output*
- *uint32\_t OutputPol*
- *uint32\_t Hysteresis*
- *uint32\_t BlankingSrce*
- *uint32\_t Mode*
- *uint32\_t WindowMode*
- *uint32\_t TriggerMode*

##### Field Documentation

- ***uint32\_t COMP\_InitTypeDef::InvertingInput***  
Selects the inverting input of the comparator. This parameter can be a value of [\*\*COMPEx\\_InvertingInput\*\*](#)
- ***uint32\_t COMP\_InitTypeDef::NonInvertingInput***  
Selects the non inverting input of the comparator. This parameter can be a value of [\*\*COMPEx\\_NonInvertingInput\*\*](#) Note: Only available on STM32F302xB/xC, STM32F303xB/xC and STM32F358xx devices
- ***uint32\_t COMP\_InitTypeDef::Output***  
Selects the output redirection of the comparator. This parameter can be a value of [\*\*COMPEx\\_Output\*\*](#)
- ***uint32\_t COMP\_InitTypeDef::OutputPol***  
Selects the output polarity of the comparator. This parameter can be a value of [\*\*COMP\\_OutputPolarity\*\*](#)
- ***uint32\_t COMP\_InitTypeDef::Hysteresis***  
Selects the hysteresis voltage of the comparator. This parameter can be a value of [\*\*COMPEx\\_Hysteresis\*\*](#) Note: Only available on STM32F302xB/xC, STM32F303xB/xC, STM32F373xB/xC, STM32F358xx and STM32F378xx devices
- ***uint32\_t COMP\_InitTypeDef::BlankingSrce***  
Selects the output blanking source of the comparator. This parameter can be a value of [\*\*COMPEx\\_BlinkingSrce\*\*](#) Note: Not available on STM32F373xB/C and STM32F378xx devices
- ***uint32\_t COMP\_InitTypeDef::Mode***  
Selects the operating consumption mode of the comparator to adjust the speed/consumption. This parameter can be a value of [\*\*COMPEx\\_Mode\*\*](#) Note: Not available on STM32F301x6/x8, STM32F302x6/x8, STM32F334x6/x8, STM32F318xx and STM32F328xx devices
- ***uint32\_t COMP\_InitTypeDef::WindowMode***  
Selects the window mode of the comparator X (X=2U, 4 or 6 if available). This parameter can be a value of [\*\*COMPEx\\_WindowMode\*\*](#)
- ***uint32\_t COMP\_InitTypeDef::TriggerMode***  
Selects the trigger mode of the comparator (interrupt mode). This parameter can be a value of [\*\*COMP\\_TriggerMode\*\*](#)

## 10.1.2 COMP\_HandleTypeDef

### Data Fields

- ***COMP\_TypeDef \* Instance***
- ***COMP\_InitTypeDef Init***
- ***HAL\_LockTypeDef Lock***
- ***\_\_IO HAL\_COMP\_StateTypeDef State***

### Field Documentation

- ***COMP\_TypeDef\* COMP\_HandleTypeDef::Instance***  
Register base address
- ***COMP\_InitTypeDef COMP\_HandleTypeDef::Init***  
COMP required parameters
- ***HAL\_LockTypeDef COMP\_HandleTypeDef::Lock***  
Locking object
- ***\_\_IO HAL\_COMP\_StateTypeDef COMP\_HandleTypeDef::State***  
COMP communication state

## 10.2 COMP Firmware driver API description

### 10.2.1 COMP Peripheral features

The STM32F3xx device family integrates up to 7 analog comparators COMP1, COMP2...COMP7:

1. The non inverting input and inverting input can be set to GPIO pins. For STM32F3xx devices please refer to the COMP peripheral section in corresponding Reference Manual.
2. The COMP output is available using HAL\_COMP\_GetOutputLevel() and can be set on GPIO pins. For STM32F3xx devices please refer to the COMP peripheral section in corresponding Reference Manual.
3. The COMP output can be redirected to embedded timers (TIM1, TIM2, TIM3...). For STM32F3xx devices please refer to the COMP peripheral section in corresponding Reference Manual.
4. Each couple of comparators COMP1 and COMP2, COMP3 and COMP4, COMP5 and COMP6 can be combined in window mode and respectively COMP1, COMP3 and COMP5 non inverting input is used as common non-inverting input.
5. The seven comparators have interrupt capability with wake-up from Sleep and Stop modes (through the EXTI controller):
  - COMP1 is internally connected to EXTI Line 21
  - COMP2 is internally connected to EXTI Line 22
  - COMP3 is internally connected to EXTI Line 29
  - COMP4 is internally connected to EXTI Line 30
  - COMP5 is internally connected to EXTI Line 31
  - COMP6 is internally connected to EXTI Line 32
  - COMP7 is internally connected to EXTI Line 33. From the corresponding IRQ handler, the right interrupt source can be retrieved with the adequate macro `__HAL_COMP_COMPx_EXTI_GET_FLAG()`.

### 10.2.2 How to use this driver

This driver provides functions to configure and program the Comparators of all STM32F3xx devices. To use the comparator, perform the following steps:

1. Fill in the HAL\_COMP\_MspInit() to
  - Configure the comparator input in analog mode using HAL\_GPIO\_Init()

- Configure the comparator output in alternate function mode using HAL\_GPIO\_Init() to map the comparator output to the GPIO pin
  - If required enable the COMP interrupt (EXTI line Interrupt): by configuring and enabling EXTI line in Interrupt mode and selecting the desired sensitivity level using HAL\_GPIO\_Init() function. After that enable the comparator interrupt vector using HAL\_NVIC\_SetPriority() and HAL\_NVIC\_EnableIRQ() functions.
2. Configure the comparator using HAL\_COMP\_Init() function:
    - Select the inverting input (input minus)
    - Select the non-inverting input (input plus)
    - Select the output polarity
    - Select the output redirection
    - Select the hysteresis level
    - Select the power mode
    - Select the event/interrupt mode HAL\_COMP\_Init() calls internally \_\_HAL\_RCC\_SYSCFG\_CLK\_ENABLE() in order to enable the comparator(s).
  3. On-the-fly reconfiguration of comparator(s) may be done by calling again HAL\_COMP\_Init() function with new input parameter values; HAL\_COMP\_MspInit() function shall be adapted to support multi configurations.
  4. Enable the comparator using HAL\_COMP\_Start() or HAL\_COMP\_Start\_IT() functions.
  5. Use HAL\_COMP\_TriggerCallback() and/or HAL\_COMP\_GetOutputLevel() functions to manage comparator outputs (events and output level).
  6. Disable the comparator using HAL\_COMP\_Stop() or HAL\_COMP\_Stop\_IT() function.
  7. De-initialize the comparator using HAL\_COMP\_DelInit() function.
  8. For safety purposes comparator(s) can be locked using HAL\_COMP\_Lock() function. Only a MCU reset can reset that protection.

### 10.2.3 Initialization and de-initialization functions

This section provides functions to initialize and de-initialize comparators.

This section contains the following APIs:

- [\*\*HAL\\_COMP\\_Init\(\)\*\*](#)
- [\*\*HAL\\_COMP\\_DelInit\(\)\*\*](#)
- [\*\*HAL\\_COMP\\_MspInit\(\)\*\*](#)
- [\*\*HAL\\_COMP\\_MspDelInit\(\)\*\*](#)

### 10.2.4 Start Stop operation functions

This section provides functions allowing to:

- Start a comparator without interrupt generation.
- Stop a comparator without interrupt generation.
- Start a comparator with interrupt generation.
- Stop a comparator with interrupt generation.
- Handle interrupts from a comparator with associated callback function.

This section contains the following APIs:

- [\*\*HAL\\_COMP\\_Start\(\)\*\*](#)
- [\*\*HAL\\_COMP\\_Stop\(\)\*\*](#)
- [\*\*HAL\\_COMP\\_Start\\_IT\(\)\*\*](#)
- [\*\*HAL\\_COMP\\_Stop\\_IT\(\)\*\*](#)
- [\*\*HAL\\_COMP\\_IRQHandler\(\)\*\*](#)
- [\*\*HAL\\_COMP\\_TriggerCallback\(\)\*\*](#)

### 10.2.5 Peripheral Control functions

This subsection provides a set of functions allowing to control the comparators.

This section contains the following APIs:

- [\*HAL\\_COMP\\_Lock\(\)\*](#)
- [\*HAL\\_COMP\\_GetOutputLevel\(\)\*](#)

### 10.2.6 Peripheral State functions

This subsection permits to get in run-time the status of the peripheral.

This section contains the following APIs:

- [\*HAL\\_COMP\\_GetState\(\)\*](#)

### 10.2.7 Detailed description of functions

#### **HAL\_COMP\_Init**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_COMP_Init (COMP_HandleTypeDef * hcomp)</b>                                                                                                                |
| Function description | Initialize the COMP peripheral according to the specified parameters in the COMP_InitTypeDef and initialize the associated handle.                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcomp:</b> COMP handle</li> </ul>                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• If the selected comparator is locked, initialization cannot be performed. To unlock the configuration, perform a system reset.</li> </ul> |

#### **HAL\_COMP\_DelInit**

|                      |                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_COMP_DelInit (COMP_HandleTypeDef * hcomp)</b>                                                                                                               |
| Function description | Deinitialize the COMP peripheral.                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcomp:</b> COMP handle</li> </ul>                                                                                                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>• If the selected comparator is locked, deinitialization cannot be performed. To unlock the configuration, perform a system reset.</li> </ul> |

#### **HAL\_COMP\_MspInit**

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <b>void HAL_COMP_MspInit (COMP_HandleTypeDef * hcomp)</b>                     |
| Function description | Initialize the COMP MSP.                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcomp:</b> COMP handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>               |

**HAL\_COMP\_MspDeInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_COMP_MspDeInit (COMP_HandleTypeDef * hcomp)</b>                 |
| Function description | DeInitialize the COMP MSP.                                                  |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcomp:</b> COMP handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>               |

**HAL\_COMP\_Start**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_COMP_Start (COMP_HandleTypeDef * hcomp)</b>        |
| Function description | Start the comparator.                                                       |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcomp:</b> COMP handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>        |

**HAL\_COMP\_Stop**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_COMP_Stop (COMP_HandleTypeDef * hcomp)</b>         |
| Function description | Stop the comparator.                                                        |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcomp:</b> COMP handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>        |

**HAL\_COMP\_Start\_IT**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_COMP_Start_IT (COMP_HandleTypeDef * hcomp)</b>     |
| Function description | Start the comparator in Interrupt mode.                                     |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcomp:</b> COMP handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status.</li></ul>       |

**HAL\_COMP\_Stop\_IT**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_COMP_Stop_IT (COMP_HandleTypeDef * hcomp)</b>      |
| Function description | Stop the comparator in Interrupt mode.                                      |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcomp:</b> COMP handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>        |

### HAL\_COMP\_IRQHandler

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_COMP_IRQHandler (COMP_HandleTypeDef * hcomp)</b>                |
| Function description | Comparator IRQ Handler.                                                     |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcomp:</b> COMP handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>        |

### HAL\_COMP\_TriggerCallback

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_COMP_TriggerCallback (COMP_HandleTypeDef * hcomp)</b>           |
| Function description | Comparator callback.                                                        |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcomp:</b> COMP handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>               |

### HAL\_COMP\_Lock

|                      |                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_COMP_Lock (COMP_HandleTypeDef * hcomp)</b>                                                  |
| Function description | Lock the selected comparator configuration.                                                                          |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcomp:</b> COMP handle</li></ul>                                          |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                 |
| Notes                | <ul style="list-style-type: none"><li>• A system reset is required to unlock the comparator configuration.</li></ul> |

### HAL\_COMP\_GetOutputLevel

|                      |                                                                      |
|----------------------|----------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_COMP_GetOutputLevel (COMP_HandleTypeDef * hcomp)</b> |
| Function description | Return the output level (high or low) of the selected comparator.    |

### HAL\_COMP\_GetState

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_COMP_StateTypeDef HAL_COMP_GetState (COMP_HandleTypeDef * hcomp)</b> |
| Function description | Return the COMP handle state.                                               |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcomp:</b> COMP handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> state</li></ul>         |

## 10.3 COMP Firmware driver defines

### 10.3.1 COMP

#### **COMP Exported Macros**

|                                           |                                                                                         |
|-------------------------------------------|-----------------------------------------------------------------------------------------|
| <code>_HAL_COMP_RESET_HANDLE_STATE</code> | <b>Description:</b>                                                                     |
|                                           | <ul style="list-style-type: none"> <li>• Reset COMP handle state.</li> </ul>            |
|                                           | <b>Parameters:</b>                                                                      |
|                                           | <ul style="list-style-type: none"> <li>• <code>_HANDLE_</code>: COMP handle.</li> </ul> |
|                                           | <b>Return value:</b>                                                                    |
|                                           | <ul style="list-style-type: none"> <li>• None</li> </ul>                                |

#### **COMP Flag**

`COMP_FLAG_LOCK` Lock flag

**COMP Extended Private macro to get the EXTI line associated with a comparator handle**

|                                 |                                                                                                            |
|---------------------------------|------------------------------------------------------------------------------------------------------------|
| <code>COMP_GET_EXTI_LINE</code> | <b>Description:</b>                                                                                        |
|                                 | <ul style="list-style-type: none"> <li>• Get the specified EXTI line for a comparator instance.</li> </ul> |
|                                 | <b>Parameters:</b>                                                                                         |
|                                 | <ul style="list-style-type: none"> <li>• <code>_INSTANCE_</code>: specifies the COMP instance.</li> </ul>  |
|                                 | <b>Return value:</b>                                                                                       |
|                                 | <ul style="list-style-type: none"> <li>• value: of</li> </ul>                                              |

**COMP Private macros to check input parameters**

`IS_COMP_OUTPUTPOL`

`IS_COMP_TRIGGERMODE`

#### **COMP Output Level**

`COMP_OUTPUTLEVEL_LOW`

`COMP_OUTPUTLEVEL_HIGH`

#### **COMP Output Polarity**

`COMP_OUTPUTPOL_NONINVERTED` COMP output on GPIO isn't inverted

`COMP_OUTPUTPOL_INVERTED` COMP output on GPIO is inverted

#### **COMP State Lock**

`COMP_STATE_BIT_LOCK`

#### **COMP Trigger Mode**

|                                    |                                         |
|------------------------------------|-----------------------------------------|
| <code>COMP_TRIGGERMODE_NONE</code> | No External Interrupt trigger detection |
|------------------------------------|-----------------------------------------|

|                                         |                                                            |
|-----------------------------------------|------------------------------------------------------------|
| <code>COMP_TRIGGERMODE_IT_RISING</code> | External Interrupt Mode with Rising edge trigger detection |
|-----------------------------------------|------------------------------------------------------------|

|                                          |                                                             |
|------------------------------------------|-------------------------------------------------------------|
| <code>COMP_TRIGGERMODE_IT_FALLING</code> | External Interrupt Mode with Falling edge trigger detection |
|------------------------------------------|-------------------------------------------------------------|

|                                       |                                                                    |
|---------------------------------------|--------------------------------------------------------------------|
| COMP_TRIGGERMODE_IT_RISING_FALLING    | External Interrupt Mode with Rising/Falling edge trigger detection |
| COMP_TRIGGERMODE_EVENT_RISING         | Event Mode with Rising edge trigger detection                      |
| COMP_TRIGGERMODE_EVENT_FALLING        | Event Mode with Falling edge trigger detection                     |
| COMP_TRIGGERMODE_EVENT_RISING_FALLING | Event Mode with Rising/Falling edge trigger detection              |

## 11 HAL COMP Extension Driver

### 11.1 COMPEx Firmware driver defines

#### 11.1.1 COMPEx

***COMP Extended Blanking Source  
(STM32F303xE/STM32F398xx/STM32F303xC/STM32F358xx Product devices)***

|                            |                                                                        |
|----------------------------|------------------------------------------------------------------------|
| COMP_BLANKINGSRCE_NONE     | No blanking source                                                     |
| COMP_BLANKINGSRCE_TIM1OC5  | TIM1 OC5 selected as blanking source for COMP1, COMP2, COMP3 and COMP7 |
| COMP_BLANKINGSRCE_TIM2OC3  | TIM2 OC5 selected as blanking source for COMP1 and COMP2               |
| COMP_BLANKINGSRCE_TIM3OC3  | TIM2 OC3 selected as blanking source for COMP1, COMP2 and COMP5        |
| COMP_BLANKINGSRCE_TIM2OC4  | TIM2 OC4 selected as blanking source for COMP3 and COMP6               |
| COMP_BLANKINGSRCE_TIM8OC5  | TIM8 OC5 selected as blanking source for COMP4, COMP5, COMP6 and COMP7 |
| COMP_BLANKINGSRCE_TIM3OC4  | TIM3 OC4 selected as blanking source for COMP4                         |
| COMP_BLANKINGSRCE_TIM15OC1 | TIM15 OC1 selected as blanking source for COMP4                        |
| COMP_BLANKINGSRCE_TIM15OC2 | TIM15 OC2 selected as blanking source for COMP6 and COMP7              |

***COMP Extended Exported Macros***

`_HAL_COMP_ENABLE`

**Description:**

- Enable the specified comparator.

**Parameters:**

- `_HANDLE_`: COMP handle.

**Return value:**

- None

`_HAL_COMP_DISABLE`

**Description:**

- Disable the specified comparator.

**Parameters:**

- `_HANDLE_`: COMP handle.

**Return value:**

- None

`__HAL_COMP_LOCK`

**Description:**

- Lock a comparator instance.

**Parameters:**

- `__HANDLE__`: COMP handle

**Return value:**

- None.

`__HAL_COMP_GET_FLAG`

**Description:**

- Check whether the specified COMP flag is set or not.

**Parameters:**

- `__HANDLE__`: COMP Handle.
- `__FLAG__`: flag to check.  
This parameter can be one of the following values:
  - `COMP_FLAG_LOCK`  
lock flag

**Return value:**

- The new state of `__FLAG__` (TRUE or FALSE).

`__HAL_COMP_COMP1_EXTI_ENABLE_RISING_EDGE`

**Description:**

- Enable the COMP1 EXTI line rising edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP1_EXTI_DISABLE_RISING_EDGE`

**Description:**

- Disable the COMP1 EXTI line rising edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP1_EXTI_ENABLE_FALLING_EDGE`

**Description:**

- Enable the COMP1 EXTI line falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP1_EXTI_DISABLE_FALLING_EDGE`

**Description:**

- Disable the COMP1 EXTI line falling edge trigger.

**Return value:**

- None

---

`__HAL_COMP_COMP1_EXTI_ENABLE_RISING_FALLING_EDGE`

**Description:**

- Enable the COMP1 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP1_EXTI_DISABLE_RISING_FALLING_EDGE`

**Description:**

- Disable the COMP1 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP1_EXTI_ENABLE_IT`

**Description:**

- Enable the COMP1 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP1_EXTI_DISABLE_IT`

**Description:**

- Disable the COMP1 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP1_EXTI_GENERATE_SWIT`

**Description:**

- Generate a software interrupt on the COMP1 EXTI line.

**Return value:**

- None

`__HAL_COMP_COMP1_EXTI_ENABLE_EVENT`

**Description:**

- Enable the COMP1 EXTI line in event mode.

**Return value:**

- None

`__HAL_COMP_COMP1_EXTI_DISABLE_EVENT`

**Description:**

- Disable the COMP1 EXTI line in event mode.

**Return value:**

- None

`__HAL_COMP_COMP1_EXTI_GET_FLAG`

**Description:**

- Check whether the COMP1 EXTI line flag is set or not.

`__HAL_COMP_COMP1_EXTI_CLEAR_FLAG`

**Return value:**

- RESET: or SET

**Description:**

- Clear the COMP1 EXTI flag.

**Return value:**

- None

**Description:**

- Enable the COMP2 EXTI line rising edge trigger.

**Return value:**

- None

**Description:**

- Disable the COMP2 EXTI line rising edge trigger.

**Return value:**

- None

**Description:**

- Enable the COMP2 EXTI line falling edge trigger.

**Return value:**

- None

**Description:**

- Disable the COMP2 EXTI line falling edge trigger.

**Return value:**

- None

**Description:**

- Enable the COMP2 EXTI line rising & falling edge trigger.

**Return value:**

- None

**Description:**

- Disable the COMP2 EXTI line rising & falling edge trigger.

**Return value:**

- None

**Description:**

- Enable the COMP2 EXTI line

in interrupt mode.

**Return value:**

- None

**Description:**

- Disable the COMP2 EXTI line in interrupt mode.

**Return value:**

- None

**Description:**

- Generate a software interrupt on the COMP2 EXTI line.

**Return value:**

- None

**Description:**

- Enable the COMP2 EXTI line in event mode.

**Return value:**

- None

**Description:**

- Disable the COMP2 EXTI line in event mode.

**Return value:**

- None

**Description:**

- Check whether the COMP2 EXTI line flag is set or not.

**Return value:**

- RESET: or SET

**Description:**

- Clear the COMP2 EXTI flag.

**Return value:**

- None

**Description:**

- Enable the COMP3 EXTI line rising edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP3_EXTI_DISABLE_RISING_EDGE`

**Description:**

- Disable the COMP3 EXTI line rising edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP3_EXTI_ENABLE_FALLING_EDGE`

**Description:**

- Enable the COMP3 EXTI line falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP3_EXTI_DISABLE_FALLING_EDGE`

**Description:**

- Disable the COMP3 EXTI line falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP3_EXTI_ENABLE_RISING_FALLING_EDGE`

**Description:**

- Enable the COMP3 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP3_EXTI_DISABLE_RISING_FALLING_EDGE`

**Description:**

- Disable the COMP3 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP3_EXTI_ENABLE_IT`

**Description:**

- Enable the COMP3 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP3_EXTI_DISABLE_IT`

**Description:**

- Disable the COMP3 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP3_EXTI_GENERATE_SWIT`

**Description:**

- Generate a software interrupt on the COMP3 EXTI line.

**Return value:**

- None

**Description:**

- Enable the COMP3 EXTI line in event mode.

**Return value:**

- None

**Description:**

- Disable the COMP3 EXTI line in event mode.

**Return value:**

- None

**Description:**

- Check whether the COMP3 EXTI line flag is set or not.

**Return value:**

- RESET: or SET

**Description:**

- Clear the COMP3 EXTI flag.

**Return value:**

- None

**Description:**

- Enable the COMP4 EXTI line rising edge trigger.

**Return value:**

- None

**Description:**

- Disable the COMP4 EXTI line rising edge trigger.

**Return value:**

- None

**Description:**

- Enable the COMP4 EXTI line falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP4_EXTI_DISABLE_FALLING_EDGE`

**Description:**

- Disable the COMP4 EXTI line falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP4_EXTI_ENABLE_RISING_FALLING_EDGE`

**Description:**

- Enable the COMP4 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP4_EXTI_DISABLE_RISING_FALLING_EDGE`

**Description:**

- Disable the COMP4 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP4_EXTI_ENABLE_IT`

**Description:**

- Enable the COMP4 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP4_EXTI_DISABLE_IT`

**Description:**

- Disable the COMP4 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP4_EXTI_GENERATE_SWIT`

**Description:**

- Generate a software interrupt on the COMP4 EXTI line.

**Return value:**

- None

`__HAL_COMP_COMP4_EXTI_ENABLE_EVENT`

**Description:**

- Enable the COMP4 EXTI line in event mode.

**Return value:**

- None

`__HAL_COMP_COMP4_EXTI_DISABLE_EVENT`

**Description:**

- Disable the COMP4 EXTI line in event mode.

**Return value:**

- None

**Description:**

- Check whether the COMP4 EXTI line flag is set or not.

**Return value:**

- RESET: or SET

**Description:**

- Clear the COMP4 EXTI flag.

**Return value:**

- None

**Description:**

- Enable the COMP5 EXTI line rising edge trigger.

**Return value:**

- None

**Description:**

- Disable the COMP5 EXTI line rising edge trigger.

**Return value:**

- None

**Description:**

- Enable the COMP5 EXTI line falling edge trigger.

**Return value:**

- None

**Description:**

- Disable the COMP5 EXTI line falling edge trigger.

**Return value:**

- None

**Description:**

- Enable the COMP5 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP5_EXTI_DISABLE_RISING_  
FALLING_EDGE`

**Description:**

- Disable the COMP5 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP5_EXTI_ENABLE_IT`

**Description:**

- Enable the COMP5 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP5_EXTI_DISABLE_IT`

**Description:**

- Disable the COMP5 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP5_EXTI_GENERATE_SWIT`

**Description:**

- Generate a software interrupt on the COMP5 EXTI line.

**Return value:**

- None

`__HAL_COMP_COMP5_EXTI_ENABLE_EVENT`

**Description:**

- Enable the COMP5 EXTI line in event mode.

**Return value:**

- None

`__HAL_COMP_COMP5_EXTI_DISABLE_EVENT`

**Description:**

- Disable the COMP5 EXTI line in event mode.

**Return value:**

- None

`__HAL_COMP_COMP5_EXTI_GET_FLAG`

**Description:**

- Check whether the COMP5 EXTI line flag is set or not.

**Return value:**

- RESET: or SET

`__HAL_COMP_COMP5_EXTI_CLEAR_FLAG`**Description:**

- Clear the COMP5 EXTI flag.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_ENABLE_RISING_EDGE`**Description:**

- Enable the COMP6 EXTI line rising edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_DISABLE_RISING_EDGE`**Description:**

- Disable the COMP6 EXTI line rising edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_ENABLE_FALLING_EDGE`**Description:**

- Enable the COMP6 EXTI line falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_DISABLE_FALLING_EDGE`**Description:**

- Disable the COMP6 EXTI line falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_ENABLE_RISING_FALLING_EDGE`**Description:**

- Enable the COMP6 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_DISABLE_RISING_FALLING_EDGE`**Description:**

- Disable the COMP6 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_ENABLE_IT`**Description:**

- Enable the COMP6 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_DISABLE_IT`**Description:**

- Disable the COMP6 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_GENERATE_SWIT`**Description:**

- Generate a software interrupt on the COMP6 EXTI line.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_ENABLE_EVENT`**Description:**

- Enable the COMP6 EXTI line in event mode.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_DISABLE_EVENT`**Description:**

- Disable the COMP6 EXTI line in event mode.

**Return value:**

- None

`__HAL_COMP_COMP6_EXTI_GET_FLAG`**Description:**

- Check whether the COMP6 EXTI line flag is set or not.

**Return value:**

- RESET: or SET

`__HAL_COMP_COMP6_EXTI_CLEAR_FLAG`**Description:**

- Clear the COMP6 EXTI flag.

**Return value:**

- None

---

`__HAL_COMP_COMP7_EXTI_ENABLE_RISING_EDGE`

**Description:**

- Enable the COMP7 EXTI line rising edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_DISABLE_RISING_EDGE`

**Description:**

- Disable the COMP7 EXTI line rising edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_ENABLE_FALLING_EDGE`

**Description:**

- Enable the COMP7 EXTI line falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_DISABLE_FALLING_EDGE`

**Description:**

- Disable the COMP7 EXTI line falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_ENABLE_RISING_FALLING_EDGE`

**Description:**

- Enable the COMP7 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_DISABLE_RISING_FALLING_EDGE`

**Description:**

- Disable the COMP7 EXTI line rising & falling edge trigger.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_ENABLE_IT`

**Description:**

- Enable the COMP7 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_DISABLE_IT`**Description:**

- Disable the COMP7 EXTI line in interrupt mode.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_GENERATE_SWIT`**Description:**

- Generate a software interrupt on the COMP7 EXTI line.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_ENABLE_EVENT`**Description:**

- Enable the COMP7 EXTI line in event mode.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_DISABLE_EVENT`**Description:**

- Disable the COMP7 EXTI line in event mode.

**Return value:**

- None

`__HAL_COMP_COMP7_EXTI_GET_FLAG`**Description:**

- Check whether the COMP7 EXTI line flag is set or not.

**Return value:**

- RESET: or SET

`__HAL_COMP_COMP7_EXTI_CLEAR_FLAG`**Description:**

- Clear the COMP7 EXTI flag.

**Return value:**

- None

***COMP Extended EXTI lines***

|                                       |                                                           |
|---------------------------------------|-----------------------------------------------------------|
| <code>COMP_EXTI_LINE_COMP1</code>     | External interrupt line 21 connected to COMP1             |
| <code>COMP_EXTI_LINE_COMP2</code>     | External interrupt line 22 connected to COMP2             |
| <code>COMP_EXTI_LINE_COMP3</code>     | External interrupt line 29 connected to COMP3             |
| <code>COMP_EXTI_LINE_COMP4</code>     | External interrupt line 30 connected to COMP4             |
| <code>COMP_EXTI_LINE_COMP5</code>     | External interrupt line 31 connected to COMP5             |
| <code>COMP_EXTI_LINE_COMP6</code>     | External interrupt line 32 connected to COMP6             |
| <code>COMP_EXTI_LINE_COMP7</code>     | External interrupt line 33 connected to COMP7             |
| <code>COMP_EXTI_LINE_REG2_MASK</code> | Mask for External interrupt line control in register xxx2 |

***COMP Extended Hysteresis***

|                        |                         |
|------------------------|-------------------------|
| COMP_HYSTERESIS_NONE   | No hysteresis           |
| COMP_HYSTERESIS_LOW    | Hysteresis level low    |
| COMP_HYSTERESIS_MEDIUM | Hysteresis level medium |
| COMP_HYSTERESIS_HIGH   | Hysteresis level high   |

***COMP Extended InvertingInput***

(STM32F302xE/STM32F303xE/STM32F398xx/STM32F302xC/STM32F303xC/STM32F358xx Product devices)

|                                |                                                                                                                                                          |
|--------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| COMP_INVERTINGINPUT_1_4VREFINT | 1U/4 VREFINT connected to comparator inverting input                                                                                                     |
| COMP_INVERTINGINPUT_1_2VREFINT | 1U/2 VREFINT connected to comparator inverting input                                                                                                     |
| COMP_INVERTINGINPUT_3_4VREFINT | 3U/4 VREFINT connected to comparator inverting input                                                                                                     |
| COMP_INVERTINGINPUT_VREFINT    | VREFINT connected to comparator inverting input                                                                                                          |
| COMP_INVERTINGINPUT_DAC1_CH1   | DAC1_CH1_OUT (PA4) connected to comparator inverting input                                                                                               |
| COMP_INVERTINGINPUT_DAC1_CH2   | DAC1_CH2_OUT (PA5) connected to comparator inverting input                                                                                               |
| COMP_INVERTINGINPUT_IO1        | IO1 (PA0 for COMP1, PA2 for COMP2, PD15 for COMP3, PE8 for COMP4, PD13 for COMP5, PD10 for COMP6, PC0 for COMP7) connected to comparator inverting input |
| COMP_INVERTINGINPUT_IO2        | IO2 (PB12 for COMP3, PB2 for COMP4, PB10 for COMP5, PB15 for COMP6) connected to comparator inverting input                                              |

COMP\_INVERTINGINPUT\_DAC1

COMP\_INVERTINGINPUT\_DAC2

***COMP Extended Private macros to check input parameters***

|                                    |
|------------------------------------|
| IS_COMP_INVERTINGINPUT             |
| IS_COMP_NONINVERTINGINPUT          |
| IS_COMP_NONINVERTINGINPUT_INSTANCE |
| IS_COMP_WINDOWMODE                 |
| IS_COMP_MODE                       |
| IS_COMP_HYSTERESIS                 |
| IS_COMP_OUTPUT                     |
| IS_COMP_OUTPUT_INSTANCE            |
| IS_COMP_BLANKINGSRCE               |
| IS_COMP_BLANKINGSRCE_INSTANCE      |

***COMP Extended miscellaneous defines***

|                             |                             |
|-----------------------------|-----------------------------|
| COMP_CSR_COMPxINSEL_MASK    | COMP_CSR_COMPxINSEL Mask    |
| COMP_CSR_COMPxOUTSEL_MASK   | COMP_CSR_COMPxOUTSEL Mask   |
| COMP_CSR_COMPxPOL_MASK      | COMP_CSR_COMPxPOL Mask      |
| COMP_CSR_RESET_VALUE        |                             |
| COMP_CSR_COMPxNONINSEL_MASK | COMP_CSR_COMPxNONINSEL mask |
| COMP_CSR_COMPxWNDWEN_MASK   | COMP_CSR_COMPxWNDWEN mask   |
| COMP_CSR_COMPxMODE_MASK     | COMP_CSR_COMPxMODE Mask     |
| COMP_CSR_COMPxHYST_MASK     | COMP_CSR_COMPxHYST Mask     |
| COMP_CSR_COMPxBLANKING_MASK | COMP_CSR_COMPxBLANKING mask |

***COMP Extended Mode***

|                         |                      |
|-------------------------|----------------------|
| COMP_MODE_HIGHSPEED     | High Speed           |
| COMP_MODE_MEDIUMSPEED   | Medium Speed         |
| COMP_MODE_LOWPOWER      | Low power mode       |
| COMP_MODE_ULTRALOWPOWER | Ultra-low power mode |

***COMP Extended NonInvertingInput (STM32F302xC/STM32F303xC/STM32F358xx Product devices)***

|                                         |                                                                                                                                                              |
|-----------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| COMP_NONINVERTINGINPUT_IO1              | IO1 (PA1 for COMP1, PA7 for COMP2, PB14 for COMP3, PB0 for COMP4, PD12 for COMP5, PD11 for COMP6, PA0 for COMP7) connected to comparator non inverting input |
| COMP_NONINVERTINGINPUT_IO2              | IO2 (PA3 for COMP2, PD14 for COMP3, PE7 for COMP4, PB13 for COMP5, PB11 for COMP6, PC1 for COMP7) connected to comparator non inverting input                |
| COMP_NONINVERTINGINPUT_DAC1SWITCHCLOSED | DAC output connected to comparator COMP1 non inverting input                                                                                                 |

***COMP Extended Output (STM32F303xC/STM32F358xx Product devices)***

|                       |                                                                         |
|-----------------------|-------------------------------------------------------------------------|
| COMP_OUTPUT_NONE      | COMP1, COMP2... or COMP7 output isn't connected to other peripherals    |
| COMP_OUTPUT_TIM1BKIN  | COMP1, COMP2... or COMP7 output connected to TIM1 Break Input (BKIN)    |
| COMP_OUTPUT_TIM1BKIN2 | COMP1, COMP2... or COMP7 output connected to TIM1 Break Input 2 (BKIN2) |
| COMP_OUTPUT_TIM8BKIN  | COMP1, COMP2... or COMP7 output connected to TIM8 Break Input (BKIN)    |
| COMP_OUTPUT_TIM8BKIN2 | COMP1, COMP2... or COMP7 output connected to TIM8 Break Input 2 (BKIN2) |

|                                 |                                                                                         |
|---------------------------------|-----------------------------------------------------------------------------------------|
| COMP_OUTPUT_TIM1BKIN2_TIM8BKIN2 | COMP1, COMP2... or COMP7 output connected to TIM1 Break Input 2 and TIM8 Break Input 2U |
| COMP_OUTPUT_TIM1OCREFCLR        | COMP1, COMP2, COMP3 or COMP7 output connected to TIM1 OCREF Clear                       |
| COMP_OUTPUT_TIM2OCREFCLR        | COMP1, COMP2 or COMP3 output connected to TIM2 OCREF Clear                              |
| COMP_OUTPUT_TIM3OCREFCLR        | COMP1, COMP2, COMP4 or COMP5 output connected to TIM3 OCREF Clear                       |
| COMP_OUTPUT_TIM8OCREFCLR        | COMP4, COMP5, COMP6 or COMP7 output connected to TIM8 OCREF Clear                       |
| COMP_OUTPUT_TIM1IC1             | COMP1 or COMP2 output connected to TIM1 Input Capture 1U                                |
| COMP_OUTPUT_TIM2IC4             | COMP1 or COMP2 output connected to TIM2 Input Capture 4U                                |
| COMP_OUTPUT_TIM3IC1             | COMP1 or COMP2 output connected to TIM3 Input Capture 1U                                |
| COMP_OUTPUT_TIM4IC1             | COMP3 output connected to TIM4 Input Capture 1U                                         |
| COMP_OUTPUT_TIM3IC2             | COMP3 output connected to TIM3 Input Capture 2U                                         |
| COMP_OUTPUT_TIM15IC1            | COMP3 output connected to TIM15 Input Capture 1U                                        |
| COMP_OUTPUT_TIM15BKIN           | COMP3 output connected to TIM15 Break Input (BKIN)                                      |
| COMP_OUTPUT_TIM3IC3             | COMP4 output connected to TIM3 Input Capture 3U                                         |
| COMP_OUTPUT_TIM15IC2            | COMP4 output connected to TIM15 Input Capture 2U                                        |
| COMP_OUTPUT_TIM4IC2             | COMP4 output connected to TIM4 Input Capture 2U                                         |
| COMP_OUTPUT_TIM15OCREFCLR       | COMP4 output connected to TIM15 OCREF Clear                                             |
| COMP_OUTPUT_TIM2IC1             | COMP5 output connected to TIM2 Input Capture 1U                                         |
| COMP_OUTPUT_TIM17IC1            | COMP5 output connected to TIM17 Input Capture 1U                                        |
| COMP_OUTPUT_TIM4IC3             | COMP5 output connected to TIM4 Input Capture 3U                                         |
| COMP_OUTPUT_TIM16BKIN           | COMP5 output connected to TIM16 Break Input (BKIN)                                      |
| COMP_OUTPUT_TIM2IC2             | COMP6 output connected to TIM2 Input Capture 2U                                         |

|                                                                                              |                                                                                                                                 |
|----------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| COMP_OUTPUT_COMP6_TIM2OCREFCLR                                                               | COMP6 output connected to TIM2 OCREF Clear                                                                                      |
| COMP_OUTPUT_TIM16OCREFCLR                                                                    | COMP6 output connected to TIM16 OCREF Clear                                                                                     |
| COMP_OUTPUT_TIM16IC1                                                                         | COMP6 output connected to TIM16 Input Capture 1U                                                                                |
| COMP_OUTPUT_TIM4IC4                                                                          | COMP6 output connected to TIM4 Input Capture 4U                                                                                 |
| COMP_OUTPUT_TIM2IC3                                                                          | COMP7 output connected to TIM2 Input Capture 3U                                                                                 |
| COMP_OUTPUT_TIM1IC2                                                                          | COMP7 output connected to TIM1 Input Capture 2U                                                                                 |
| COMP_OUTPUT_TIM17OCREFCLR                                                                    | COMP7 output connected to TIM17 OCREF Clear                                                                                     |
| COMP_OUTPUT_TIM17BKIN                                                                        | COMP7 output connected to TIM17 Break Input (BKIN)                                                                              |
| <b><i>COMP Extended WindowMode (STM32F302xC/STM32F303xC/STM32F358xx Product devices)</i></b> |                                                                                                                                 |
| COMP_WINDOWMODE_DISABLE                                                                      | Window mode disabled                                                                                                            |
| COMP_WINDOWMODE_ENABLE                                                                       | Window mode enabled: non inverting input of comparator X (x=2U,4,6U) is connected to the non inverting input of comparator X-1U |

## 12 HAL CORTEX Generic Driver

### 12.1 CORTEX Firmware driver registers structures

#### 12.1.1 MPU\_Region\_InitTypeDef

##### Data Fields

- *uint8\_t Enable*
- *uint8\_t Number*
- *uint32\_t BaseAddress*
- *uint8\_t Size*
- *uint8\_t SubRegionDisable*
- *uint8\_t TypeExtField*
- *uint8\_t AccessPermission*
- *uint8\_t DisableExec*
- *uint8\_t IsShareable*
- *uint8\_t IsCacheable*
- *uint8\_t IsBufferable*

##### Field Documentation

- ***uint8\_t MPU\_Region\_InitTypeDef::Enable***  
Specifies the status of the region. This parameter can be a value of [\*\*CORTEX MPU Region Enable\*\*](#)
- ***uint8\_t MPU\_Region\_InitTypeDef::Number***  
Specifies the number of the region to protect. This parameter can be a value of [\*\*CORTEX MPU Region Number\*\*](#)
- ***uint32\_t MPU\_Region\_InitTypeDef::BaseAddress***  
Specifies the base address of the region to protect.
- ***uint8\_t MPU\_Region\_InitTypeDef::Size***  
Specifies the size of the region to protect. This parameter can be a value of [\*\*CORTEX MPU Region Size\*\*](#)
- ***uint8\_t MPU\_Region\_InitTypeDef::SubRegionDisable***  
Specifies the number of the subregion protection to disable. This parameter must be a number between Min\_Data = 0x00 and Max\_Data = 0xFF
- ***uint8\_t MPU\_Region\_InitTypeDef::TypeExtField***  
Specifies the TEX field level. This parameter can be a value of [\*\*CORTEX MPU TEX Levels\*\*](#)
- ***uint8\_t MPU\_Region\_InitTypeDef::AccessPermission***  
Specifies the region access permission type. This parameter can be a value of [\*\*CORTEX MPU Region Permission Attributes\*\*](#)
- ***uint8\_t MPU\_Region\_InitTypeDef::DisableExec***  
Specifies the instruction access status. This parameter can be a value of [\*\*CORTEX MPU Instruction Access\*\*](#)
- ***uint8\_t MPU\_Region\_InitTypeDef::IsShareable***  
Specifies the shareability status of the protected region. This parameter can be a value of [\*\*CORTEX MPU Access Shareable\*\*](#)
- ***uint8\_t MPU\_Region\_InitTypeDef::IsCacheable***  
Specifies the cacheable status of the region protected. This parameter can be a value of [\*\*CORTEX MPU Access Cacheable\*\*](#)

- ***uint8\_t MPU\_Region\_InitTypeDef::IsBufferable***  
Specifies the bufferable status of the protected region. This parameter can be a value of **CORTEX\_MPUMemoryAccessBufferable**

## 12.2 CORTEX Firmware driver API description

### 12.2.1 How to use this driver

#### How to configure Interrupts using CORTEX HAL driver

This section provides functions allowing to configure the NVIC interrupts (IRQ). The Cortex-M4 exceptions are managed by CMSIS functions.

1. Configure the NVIC Priority Grouping using HAL\_NVIC\_SetPriorityGrouping() function
2. Configure the priority of the selected IRQ Channels using HAL\_NVIC\_SetPriority()
3. Enable the selected IRQ Channels using HAL\_NVIC\_EnableIRQ() When the NVIC\_PRIORITYGROUP\_0 is selected, IRQ pre-emption is no more possible. The pending IRQ priority will be managed only by the sub priority. IRQ priority order (sorted by highest to lowest priority): Lowest pre-emption priorityLowest sub priorityLowest hardware priority (IRQ number)

#### How to configure Systick using CORTEX HAL driver

Setup SysTick Timer for time base

- The HAL\_SYSTICK\_Config() function calls the SysTick\_Config() function which is a CMSIS function that:
  - Configures the SysTick Reload register with value passed as function parameter.
  - Configures the SysTick IRQ priority to the lowest value (0x0FU).
  - Resets the SysTick Counter register.
  - Configures the SysTick Counter clock source to be Core Clock Source (HCLK).
  - Enables the SysTick Interrupt.
  - Starts the SysTick Counter.
- You can change the SysTick Clock source to be HCLK\_Div8 by calling the macro \_\_HAL\_CORTEX\_SYSTICKCLK\_CONFIG(SYSTICK\_CLKSOURCE\_HCLK\_DIV8) just after the HAL\_SYSTICK\_Config() function call. The \_\_HAL\_CORTEX\_SYSTICKCLK\_CONFIG() macro is defined inside the stm32f3xx\_hal\_cortex.h file.
- You can change the SysTick IRQ priority by calling the HAL\_NVIC\_SetPriority(SysTick\_IRQn,...) function just after the HAL\_SYSTICK\_Config() function call. The HAL\_NVIC\_SetPriority() call the NVIC\_SetPriority() function which is a CMSIS function.
- To adjust the SysTick time base, use the following formula: Reload Value = SysTick Counter Clock (Hz) x Desired Time base (s)
  - Reload Value is the parameter to be passed for HAL\_SYSTICK\_Config() function
  - Reload Value should not exceed 0xFFFFFFF

### 12.2.2 Initialization and de-initialization functions

This section provides the CORTEX HAL driver functions allowing to configure Interrupts Systick functionalities

This section contains the following APIs:

- [\*HAL\\_NVIC\\_SetPriorityGrouping\(\)\*](#)
- [\*HAL\\_NVIC\\_SetPriority\(\)\*](#)
- [\*HAL\\_NVIC\\_EnableIRQ\(\)\*](#)
- [\*HAL\\_NVIC\\_DisableIRQ\(\)\*](#)
- [\*HAL\\_NVIC\\_SystemReset\(\)\*](#)
- [\*HAL\\_SYSTICK\\_Config\(\)\*](#)

### 12.2.3 Peripheral Control functions

This subsection provides a set of functions allowing to control the CORTEX (NVIC, SYSTICK, MPU) functionalities.

This section contains the following APIs:

- [\*HAL\\_MPU\\_Disable\(\)\*](#)
- [\*HAL\\_MPU\\_Enable\(\)\*](#)
- [\*HAL\\_MPU\\_ConfigRegion\(\)\*](#)
- [\*HAL\\_NVIC\\_GetPriorityGrouping\(\)\*](#)
- [\*HAL\\_NVIC\\_GetPriority\(\)\*](#)
- [\*HAL\\_NVIC\\_SetPendingIRQ\(\)\*](#)
- [\*HAL\\_NVIC\\_GetPendingIRQ\(\)\*](#)
- [\*HAL\\_NVIC\\_ClearPendingIRQ\(\)\*](#)
- [\*HAL\\_NVIC\\_GetActive\(\)\*](#)
- [\*HAL\\_SYSTICK\\_CLKSourceConfig\(\)\*](#)
- [\*HAL\\_SYSTICK\\_IRQHandler\(\)\*](#)
- [\*HAL\\_SYSTICK\\_Callback\(\)\*](#)

### 12.2.4 Detailed description of functions

#### **HAL\_NVIC\_SetPriorityGrouping**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_NVIC_SetPriorityGrouping (uint32_t PriorityGroup)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Sets the priority grouping field (pre-emption priority and subpriority) using the required unlock sequence.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>PriorityGroup:</b> The priority grouping bits length. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- NVIC_PRIORITYGROUP_0: 0 bits for pre-emption priority 4 bits for subpriority</li> <li>- NVIC_PRIORITYGROUP_1: 1 bits for pre-emption priority 3 bits for subpriority</li> <li>- NVIC_PRIORITYGROUP_2: 2 bits for pre-emption priority 2 bits for subpriority</li> <li>- NVIC_PRIORITYGROUP_3: 3 bits for pre-emption priority 1 bits for subpriority</li> <li>- NVIC_PRIORITYGROUP_4: 4 bits for pre-emption priority 0 bits for subpriority</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Notes                | <ul style="list-style-type: none"> <li>• When the NVIC_PriorityGroup_0 is selected, IRQ pre-emption is no more possible. The pending IRQ priority will be managed only by the subpriority.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

**HAL\_NVIC\_SetPriority**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_NVIC_SetPriority (IRQn_Type IRQn, uint32_t PreemptPriority, uint32_t SubPriority)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Sets the priority of an interrupt.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>IRQn:</b> External interrupt number This parameter can be an enumerator of IRQn_Type enumeration (For the complete STM32 Devices IRQ Channels list, please refer to the appropriate CMSIS device file (stm32f3xxxx.h))</li> <li>• <b>PreemptPriority:</b> The pre-emption priority for the IRQn channel. This parameter can be a value between 0 and 15 as described in the table CORTEX_NVIC_Priority_Table A lower priority value indicates a higher priority</li> <li>• <b>SubPriority:</b> the subpriority level for the IRQ channel. This parameter can be a value between 0 and 15 as described in the table CORTEX_NVIC_Priority_Table A lower priority value indicates a higher priority.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

**HAL\_NVIC\_EnableIRQ**

|                      |                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_NVIC_EnableIRQ (IRQn_Type IRQn)</b>                                                                                                                                                                                                                               |
| Function description | Enables a device specific interrupt in the NVIC interrupt controller.                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>IRQn:</b> External interrupt number This parameter can be an enumerator of IRQn_Type enumeration (For the complete STM32 Devices IRQ Channels list, please refer to the appropriate CMSIS device file (stm32f3xxxx.h))</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>• To configure interrupts priority correctly, the NVIC_PriorityGroupConfig() function should be called before.</li> </ul>                                                                                                              |

**HAL\_NVIC\_DisableIRQ**

|                      |                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_NVIC_DisableIRQ (IRQn_Type IRQn)</b>                                                                                                                                                                                                                              |
| Function description | Disables a device specific interrupt in the NVIC interrupt controller.                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>IRQn:</b> External interrupt number This parameter can be an enumerator of IRQn_Type enumeration (For the complete STM32 Devices IRQ Channels list, please refer to the appropriate CMSIS device file (stm32f3xxxx.h))</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                               |

**HAL\_NVIC\_SystemReset**

|                      |                                                                 |
|----------------------|-----------------------------------------------------------------|
| Function name        | <b>void HAL_NVIC_SystemReset (void )</b>                        |
| Function description | Initiates a system reset request to reset the MCU.              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |

**HAL\_SYSTICK\_Config**

|                      |                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_SYSTICK_Config (uint32_t TicksNumb)</b>                                                                           |
| Function description | Initializes the System Timer and its interrupt, and starts the System Tick Timer.                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TicksNumb:</b> Specifies the ticks Number of ticks between two interrupts.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>status:</b> - 0 Function succeeded.</li> <li>– 1 Function failed.</li> </ul>          |

**HAL MPU ConfigRegion**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL MPU_ConfigRegion (MPU_Region_InitTypeDef * MPU_Init)</b>                                                                                                               |
| Function description | Initializes and configures the Region and the memory to be protected.                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>MPU_Init:</b> Pointer to a MPU_Region_InitTypeDef structure that contains the initialization and configuration information.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                    |

**HAL\_NVIC\_GetPriorityGrouping**

|                      |                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_NVIC_GetPriorityGrouping (void )</b>                                                                      |
| Function description | Gets the priority grouping field from the NVIC Interrupt Controller.                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Priority:</b> grouping field (SCB-&gt;AIRCR [10:8] PRIGROUP field)</li> </ul> |

**HAL\_NVIC\_GetPriority**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_NVIC_GetPriority (IRQn_Type IRQn, uint32_t PriorityGroup, uint32_t * pPreemptPriority, uint32_t * pSubPriority)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Gets the priority of an interrupt.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>IRQn:</b> External interrupt number This parameter can be an enumerator of IRQn_Type enumeration (For the complete STM32 Devices IRQ Channels list, please refer to the appropriate CMSIS device file (stm32f3xxxx.h))</li> <li>• <b>PriorityGroup:</b> the priority grouping bits length. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– NVIC_PRIORITYGROUP_0: 0 bits for pre-emption priority 4 bits for subpriority</li> <li>– NVIC_PRIORITYGROUP_1: 1 bits for pre-emption priority 3 bits for subpriority</li> <li>– NVIC_PRIORITYGROUP_2: 2 bits for pre-emption priority 2 bits for subpriority</li> <li>– NVIC_PRIORITYGROUP_3: 3 bits for pre-emption priority 1 bits for subpriority</li> <li>– NVIC_PRIORITYGROUP_4: 4 bits for pre-emption priority 0 bits for subpriority</li> </ul> </li> </ul> |

- **pPreemptPriority:** Pointer on the Preemptive priority value (starting from 0).
  - **pSubPriority:** Pointer on the Subpriority value (starting from 0).
- Return values**
- **None**

### **HAL\_NVIC\_GetPendingIRQ**

|                      |                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_NVIC_GetPendingIRQ (IRQn_Type IRQn)</b>                                                                                                                                                                                                                       |
| Function description | Gets Pending Interrupt (reads the pending register in the NVIC and returns the pending bit for the specified interrupt).                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>IRQn:</b> External interrupt number This parameter can be an enumerator of IRQn_Type enumeration (For the complete STM32 Devices IRQ Channels list, please refer to the appropriate CMSIS device file (stm32f3xxxx.h))</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>status:</b> - 0 Interrupt status is not pending.</li> <li>– 1 Interrupt status is pending.</li> </ul>                                                                                                                             |

### **HAL\_NVIC\_SetPendingIRQ**

|                      |                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_NVIC_SetPendingIRQ (IRQn_Type IRQn)</b>                                                                                                                                                                                                                           |
| Function description | Sets Pending bit of an external interrupt.                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>IRQn:</b> External interrupt number This parameter can be an enumerator of IRQn_Type enumeration (For the complete STM32 Devices IRQ Channels list, please refer to the appropriate CMSIS device file (stm32f3xxxx.h))</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                               |

### **HAL\_NVIC\_ClearPendingIRQ**

|                      |                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_NVIC_ClearPendingIRQ (IRQn_Type IRQn)</b>                                                                                                                                                                                                                         |
| Function description | Clears the pending bit of an external interrupt.                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>IRQn:</b> External interrupt number This parameter can be an enumerator of IRQn_Type enumeration (For the complete STM32 Devices IRQ Channels list, please refer to the appropriate CMSIS device file (stm32f3xxxx.h))</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                               |

### **HAL\_NVIC\_GetActive**

|                      |                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_NVIC_GetActive (IRQn_Type IRQn)</b>                                                                                                                                                                                                                           |
| Function description | Gets active interrupt ( reads the active register in NVIC and returns the active bit).                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>IRQn:</b> External interrupt number This parameter can be an enumerator of IRQn_Type enumeration (For the complete STM32 Devices IRQ Channels list, please refer to the appropriate CMSIS device file (stm32f3xxxx.h))</li> </ul> |

- Return values
- **status:** - 0 Interrupt status is not pending.  
– 1 Interrupt status is pending.

### **HAL\_SYSTICK\_CLKSourceConfig**

Function name **void HAL\_SYSTICK\_CLKSourceConfig (uint32\_t CLKSource)**

Function description Configures the SysTick clock source.

- Parameters
- **CLKSource:** specifies the SysTick clock source. This parameter can be one of the following values:
    - SYSTICK\_CLKSOURCE\_HCLK\_DIV8: AHB clock divided by 8 selected as SysTick clock source.
    - SYSTICK\_CLKSOURCE\_HCLK: AHB clock selected as SysTick clock source.

- Return values
- **None**

### **HAL\_SYSTICK\_IRQHandler**

Function name **void HAL\_SYSTICK\_IRQHandler (void )**

Function description This function handles SYSTICK interrupt request.

- Return values
- **None**

### **HAL\_SYSTICK\_Callback**

Function name **void HAL\_SYSTICK\_Callback (void )**

Function description SYSTICK callback.

- Return values
- **None**

### **HAL\_MPU\_Disable**

Function name **void HAL\_MPU\_Disable (void )**

Function description Disables the MPU also clears the HFNMIEA bit (ARM recommendation)

- Return values
- **None**

### **HAL\_MPU\_Enable**

Function name **void HAL\_MPU\_Enable (uint32\_t MPU\_Control)**

Function description Enables the MPU.

- Parameters
- **MPU\_Control:** Specifies the control mode of the MPU during hard fault, NMI, FAULTMASK and privileged access to the default memory This parameter can be one of the following values:
    - MPU\_HFNMI\_PRIVDEF\_NONE
    - MPU\_HARDFAULT\_NMI
    - MPU\_PRIVILEGED\_DEFAULT
    - MPU\_HFNMI\_PRIVDEF

- Return values
- **None**

## 12.3 CORTEX Firmware driver defines

### 12.3.1 CORTEX

#### **CORTEX MPU Instruction Access Bufferable**

MPU\_ACCESS\_BUFFERABLE

MPU\_ACCESS\_NOT\_BUFFERABLE

#### **CORTEX MPU Instruction Access Cacheable**

MPU\_ACCESS\_CACHEABLE

MPU\_ACCESS\_NOT\_CACHEABLE

#### **CORTEX MPU Instruction Access Shareable**

MPU\_ACCESS\_SHAREABLE

MPU\_ACCESS\_NOT\_SHAREABLE

#### **MPU\_HFNMI and PRIVILEGED Access control**

MPU\_HFNMI\_PRIVDEF\_NONE

MPU\_HARDFAULT\_NMI

MPU\_PRIVILEGED\_DEFAULT

MPU\_HFNMI\_PRIVDEF

#### **CORTEX MPU Instruction Access**

MPU\_INSTRUCTION\_ACCESS\_ENABLE

MPU\_INSTRUCTION\_ACCESS\_DISABLE

#### **CORTEX MPU Region Enable**

MPU\_REGION\_ENABLE

MPU\_REGION\_DISABLE

#### **CORTEX MPU Region Number**

MPU\_REGION\_NUMBER0

MPU\_REGION\_NUMBER1

MPU\_REGION\_NUMBER2

MPU\_REGION\_NUMBER3

MPU\_REGION\_NUMBER4

MPU\_REGION\_NUMBER5

MPU\_REGION\_NUMBER6

MPU\_REGION\_NUMBER7

#### **CORTEX MPU Region Permission Attributes**

MPU\_REGION\_NO\_ACCESS

MPU\_REGION\_PRIV\_RW

MPU\_REGION\_PRIV\_RW\_URO

MPU\_REGION\_FULL\_ACCESS

MPU\_REGION\_PRIV\_RO  
MPU\_REGION\_PRIV\_RO\_URO

**CORTEX MPU Region Size**

MPU\_REGION\_SIZE\_32B  
MPU\_REGION\_SIZE\_64B  
MPU\_REGION\_SIZE\_128B  
MPU\_REGION\_SIZE\_256B  
MPU\_REGION\_SIZE\_512B  
MPU\_REGION\_SIZE\_1KB  
MPU\_REGION\_SIZE\_2KB  
MPU\_REGION\_SIZE\_4KB  
MPU\_REGION\_SIZE\_8KB  
MPU\_REGION\_SIZE\_16KB  
MPU\_REGION\_SIZE\_32KB  
MPU\_REGION\_SIZE\_64KB  
MPU\_REGION\_SIZE\_128KB  
MPU\_REGION\_SIZE\_256KB  
MPU\_REGION\_SIZE\_512KB  
MPU\_REGION\_SIZE\_1MB  
MPU\_REGION\_SIZE\_2MB  
MPU\_REGION\_SIZE\_4MB  
MPU\_REGION\_SIZE\_8MB  
MPU\_REGION\_SIZE\_16MB  
MPU\_REGION\_SIZE\_32MB  
MPU\_REGION\_SIZE\_64MB  
MPU\_REGION\_SIZE\_128MB  
MPU\_REGION\_SIZE\_256MB  
MPU\_REGION\_SIZE\_512MB  
MPU\_REGION\_SIZE\_1GB  
MPU\_REGION\_SIZE\_2GB  
MPU\_REGION\_SIZE\_4GB

**MPU TEX Levels**

MPU\_TEX\_LEVEL0  
MPU\_TEX\_LEVEL1  
MPU\_TEX\_LEVEL2

**CORTEX Preemption Priority Group**

NVIC\_PRIORITYGROUP\_0 0 bits for pre-emption priority 4 bits for subpriority

NVIC\_PRIORITYGROUP\_1 1 bits for pre-emption priority 3 bits for subpriority

NVIC\_PRIORITYGROUP\_2 2 bits for pre-emption priority 2 bits for subpriority

NVIC\_PRIORITYGROUP\_3 3 bits for pre-emption priority 1 bits for subpriority

NVIC\_PRIORITYGROUP\_4 4 bits for pre-emption priority 0 bits for subpriority

**CORTEX SysTick clock source**

SYSTICK\_CLKSOURCE\_HCLK\_DIV8

SYSTICK\_CLKSOURCE\_HCLK

## 13 HAL CRC Generic Driver

### 13.1 CRC Firmware driver registers structures

#### 13.1.1 CRC\_InitTypeDef

##### Data Fields

- *uint8\_t DefaultPolynomialUse*
- *uint8\_t DefaultInitValueUse*
- *uint32\_t GeneratingPolynomial*
- *uint32\_t CRCLength*
- *uint32\_t InitValue*
- *uint32\_t InputDataInversionMode*
- *uint32\_t OutputDataInversionMode*

##### Field Documentation

- ***uint8\_t CRC\_InitTypeDef::DefaultPolynomialUse***  
This parameter is a value of [CRC\\_Default\\_Polynomial](#) and indicates if default polynomial is used. If set to DEFAULT\_POLYNOMIAL\_ENABLE, resort to default  $X^{32}U + X^{26}U + X^{23}U + X^{22}U + X^{16}U + X^{12}U + X^{11}U + X^{10}U + X^8U + X^7U + X^5U + X^4U + X^2U + X + 1$ . In that case, there is no need to set GeneratingPolynomial field. If otherwise set to DEFAULT\_POLYNOMIAL\_DISABLE, GeneratingPolynomial and CRCLength fields must be set.
- ***uint8\_t CRC\_InitTypeDef::DefaultInitValueUse***  
This parameter is a value of [CRC\\_Default\\_InitValue\\_Use](#) and indicates if default init value is used. If set to DEFAULT\_INIT\_VALUE\_ENABLE, resort to default 0xFFFFFFFF value. In that case, there is no need to set InitValue field. If otherwise set to DEFAULT\_INIT\_VALUE\_DISABLE, InitValue field must be set.
- ***uint32\_t CRC\_InitTypeDef::GeneratingPolynomial***  
Set CRC generating polynomial as a 7U, 8U, 16 or 32-bit long value for a polynomial degree respectively equal to 7U, 8U, 16 or 32. This field is written in normal representation, e.g., for a polynomial of degree 7U,  $X^7U + X^6U + X^5U + X^2U + 1$  is written 0x65. No need to specify it if DefaultPolynomialUse is set to DEFAULT\_POLYNOMIAL\_ENABLE.
- ***uint32\_t CRC\_InitTypeDef::CRCLength***  
This parameter is a value of [CRC\\_Polynomial\\_Sizes](#) and indicates CRC length. Value can be either one of CRC\_POLYLENGTH\_32B (32-bit CRC), CRC\_POLYLENGTH\_16B (16-bit CRC), CRC\_POLYLENGTH\_8B (8-bit CRC), CRC\_POLYLENGTH\_7B (7-bit CRC).
- ***uint32\_t CRC\_InitTypeDef::InitValue***  
Init value to initiate CRC computation. No need to specify it if DefaultInitValueUse is set to DEFAULT\_INIT\_VALUE\_ENABLE.
- ***uint32\_t CRC\_InitTypeDef::InputDataInversionMode***  
This parameter is a value of [CRCEX\\_Input\\_Data\\_Inversion](#) and specifies input data inversion mode. Can be either one of the following values  
CRC\_INPUTDATA\_INVERSION\_NONE, no input data inversion  
CRC\_INPUTDATA\_INVERSION\_BYTE, byte-wise inversion, 0x1A2B3C4D becomes 0x58D43CB2  
CRC\_INPUTDATA\_INVERSION\_HALFWORD, halfword-wise inversion, 0x1A2B3C4D becomes 0xD458B23C  
CRC\_INPUTDATA\_INVERSION\_WORD, word-wise inversion, 0x1A2B3C4D becomes 0xB23CD458U

- ***uint32\_t CRC\_InitTypeDef::OutputDataInversionMode***  
This parameter is a value of [\*\*CRCEx\\_Output\\_Data\\_Inversion\*\*](#) and specifies output data (i.e. CRC) inversion mode. Can be either  
CRC\_OUTPUTDATA\_INVERSION\_DISABLE: no CRC inversion,  
CRC\_OUTPUTDATA\_INVERSION\_ENABLE: CRC 0x11223344 is converted into 0x22CC4488U

### 13.1.2 CRC\_HandleTypeDef

#### Data Fields

- ***CRC\_TypeDef \* Instance***
- ***CRC\_InitTypeDef Init***
- ***HAL\_LockTypeDef Lock***
- ***\_IO HAL\_CRC\_StateTypeDef State***
- ***uint32\_t InputDataFormat***

#### Field Documentation

- ***CRC\_TypeDef\* CRC\_HandleTypeDef::Instance***  
Register base address
- ***CRC\_InitTypeDef CRC\_HandleTypeDef::Init***  
CRC configuration parameters
- ***HAL\_LockTypeDef CRC\_HandleTypeDef::Lock***  
CRC Locking object
- ***\_IO HAL\_CRC\_StateTypeDef CRC\_HandleTypeDef::State***  
CRC communication state
- ***uint32\_t CRC\_HandleTypeDef::InputDataFormat***  
This parameter is a value of [\*\*CRC\\_Input\\_Buffer\\_Format\*\*](#) and specifies input data format. Can be either CRC\_INPUTDATA\_FORMAT\_BYTES, input data is a stream of bytes (8-bit data) CRC\_INPUTDATA\_FORMAT\_HALFWORDS, input data is a stream of half-words (16-bit data) CRC\_INPUTDATA\_FORMAT\_WORDS, input data is a stream of words (32-bit data) Note that constant CRC\_INPUT\_FORMAT\_UNDEFINED is defined but an initialization error must occur if InputBufferFormat is not one of the three values listed above

## 13.2 CRC Firmware driver API description

### 13.2.1 How to use this driver

- Enable CRC AHB clock using `__HAL_RCC_CRC_CLK_ENABLE();`
- Initialize CRC calculator
  - specify generating polynomial (IP default or non-default one)
  - specify initialization value (IP default or non-default one)
  - specify input data format
  - specify input or output data inversion mode if any
- Use `HAL_CRC_Accumulate()` function to compute the CRC value of the input data buffer starting with the previously computed CRC as initialization value
- Use `HAL_CRC_Calculate()` function to compute the CRC value of the input data buffer starting with the defined initialization value (default or non-default) to initiate CRC calculation

### 13.2.2 Initialization and de-initialization functions

This section provides functions allowing to:

- Initialize the CRC according to the specified parameters in the CRC\_InitTypeDef and create the associated handle
- DeInitialize the CRC peripheral
- Initialize the CRC MSP (MCU Specific Package)
- DeInitialize the CRC MSP

This section contains the following APIs:

- [\*HAL\\_CRC\\_Init\(\)\*](#)
- [\*HAL\\_CRC\\_DelInit\(\)\*](#)
- [\*HAL\\_CRC\\_MspInit\(\)\*](#)
- [\*HAL\\_CRC\\_MspDelInit\(\)\*](#)

### 13.2.3 Peripheral Control functions

This section provides functions allowing to:

- compute the 7U, 8U, 16 or 32-bit CRC value of an 8U, 16 or 32-bit data buffer using the combination of the previous CRC value and the new one  
or
- compute the 7U, 8U, 16 or 32-bit CRC value of an 8U, 16 or 32-bit data buffer independently of the previous CRC value.

This section contains the following APIs:

- [\*HAL\\_CRC\\_Accumulate\(\)\*](#)
- [\*HAL\\_CRC\\_Calculate\(\)\*](#)

### 13.2.4 Peripheral State functions

This subsection permits to get in run-time the status of the peripheral.

This section contains the following APIs:

- [\*HAL\\_CRC\\_GetState\(\)\*](#)

### 13.2.5 Detailed description of functions

#### **HAL\_CRC\_Init**

|                      |                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CRC_Init (CRC_HandleTypeDef *<br/>hcrc)</b>                                                     |
| Function description | Initialize the CRC according to the specified parameters in the<br>CRC_InitTypeDef and initialize the associated handle. |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hcrc:</b> CRC handle</li></ul>                                                |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                     |

**HAL\_CRC\_DeInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CRC_DeInit (CRC_HandleTypeDef * hcrc)</b>          |
| Function description | DeInitialize the CRC peripheral.                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcrc:</b> CRC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>      |

**HAL\_CRC\_MspInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_CRC_MspInit (CRC_HandleTypeDef * hcrc)</b>                      |
| Function description | Initializes the CRC MSP.                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcrc:</b> CRC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_CRC\_MspDeInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_CRC_MspDeInit (CRC_HandleTypeDef * hcrc)</b>                    |
| Function description | DeInitialize the CRC MSP.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcrc:</b> CRC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_CRC\_Accumulate**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_CRC_Accumulate (CRC_HandleTypeDef * hcrc, uint32_t pBuffer, uint32_t BufferLength)</b>                                                                                                                                                                                                                                                                                                                |
| Function description | Compute the 7, 8, 16 or 32-bit CRC value of an 8, 16 or 32-bit data buffer starting with the previously computed CRC as initialization value.                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcrc:</b> CRC handle</li> <li>• <b>pBuffer:</b> pointer to the input data buffer, exact input data format is provided by hcrc-&gt;InputDataFormat.</li> <li>• <b>BufferLength:</b> input data buffer length (number of bytes if pBuffer type is * uint8_t, number of half-words if pBuffer type is * uint16_t, number of words if pBuffer type is * uint32_t).</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>uint32_t:</b> CRC (returned value LSBs for CRC shorter than 32 bits)</li> </ul>                                                                                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• By default, the API expects a uint32_t pointer as input buffer parameter. Input buffer pointers with other types simply need to be cast in uint32_t and the API will internally adjust its input data processing based on the handle field hcrc-&gt;InputDataFormat.</li> </ul>                                                                                              |

**HAL\_CRC\_Calculate**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_CRC_Calculate (CRC_HandleTypeDef * hcrc,<br/>uint32_t pBuffer, uint32_t BufferLength)</code>                                                                                                                                                                                                                                                                                                       |
| Function description | Compute the 7, 8, 16 or 32-bit CRC value of an 8, 16 or 32-bit data buffer starting with hcrc->Instance->INIT as initialization value.                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcrc:</b> CRC handle</li> <li>• <b>pBuffer:</b> pointer to the input data buffer, exact input data format is provided by hcrc-&gt;InputDataFormat.</li> <li>• <b>BufferLength:</b> input data buffer length (number of bytes if pBuffer type is * uint8_t, number of half-words if pBuffer type is * uint16_t, number of words if pBuffer type is * uint32_t).</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>uint32_t:</b> CRC (returned value LSBs for CRC shorter than 32 bits)</li> </ul>                                                                                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• By default, the API expects a uint32_t pointer as input buffer parameter. Input buffer pointers with other types simply need to be cast in uint32_t and the API will internally adjust its input data processing based on the handle field hcrc-&gt;InputDataFormat.</li> </ul>                                                                                              |

**HAL\_CRC\_GetState**

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>HAL_CRC_StateTypeDef HAL_CRC_GetState<br/>(CRC_HandleTypeDef * hcrc)</code> |
| Function description | Return the CRC handle state.                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcrc:</b> CRC handle</li> </ul>       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>             |

## 13.3 CRC Firmware driver defines

### 13.3.1 CRC

***Default CRC computation initialization value***

`DEFAULT_CRC_INITVALUE` Initial CRC default value

***Indicates whether or not default init value is used***

`DEFAULT_INIT_VALUE_ENABLE` Enable initial CRC default value

`DEFAULT_INIT_VALUE_DISABLE` Disable initial CRC default value

***Indicates whether or not default polynomial is used***

`DEFAULT_POLYNOMIAL_ENABLE` Enable default generating polynomial 0x04C11DB7

`DEFAULT_POLYNOMIAL_DISABLE` Disable default generating polynomial 0x04C11DB7U

***Default CRC generating polynomial***

`DEFAULT_CRC32_POLY`  $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$

**CRC Exported Macros**`__HAL_CRC_RESET_HANDLE_STATE`**Description:**

- Reset CRC handle state.

**Parameters:**

- `__HANDLE__`: CRC handle.

**Return value:**

- None

`__HAL_CRC_DR_RESET`**Description:**

- Reset CRC Data Register.

**Parameters:**

- `__HANDLE__`: CRC handle

**Return value:**

- None

`__HAL_CRC_INITIALCRCVALUE_CONFIG`**Description:**

- Set CRC INIT non-default value.

**Parameters:**

- `__HANDLE__`: CRC handle
- `__INIT__`: 32-bit initial value

**Return value:**

- None

`__HAL_CRC_SET_IDR`**Description:**

- Store a 8-bit data in the Independent Data(ID) register.

**Parameters:**

- `__HANDLE__`: CRC handle
- `__VALUE__`: 8-bit value to be stored in the ID register

**Return value:**

- None

`__HAL_CRC_GET_IDR`**Description:**

- Return the 8-bit data stored in the Independent Data(ID) register.

**Parameters:**

- `__HANDLE__`: CRC handle

**Return value:**

- 8-bit: value of the ID register

***Input Buffer Format***

|                                |                                |
|--------------------------------|--------------------------------|
| CRC_INPUTDATA_FORMAT_UNDEFINED | Undefined input data format    |
| CRC_INPUTDATA_FORMAT_BYTES     | Input data in byte format      |
| CRC_INPUTDATA_FORMAT_HALFWORDS | Input data in half-word format |
| CRC_INPUTDATA_FORMAT_WORDS     | Input data in word format      |

***Polynomial sizes to configure the IP***

|                    |                                               |
|--------------------|-----------------------------------------------|
| CRC_POLYLENGTH_32B | Resort to a 32-bit long generating polynomial |
| CRC_POLYLENGTH_16B | Resort to a 16-bit long generating polynomial |
| CRC_POLYLENGTH_8B  | Resort to a 8-bit long generating polynomial  |
| CRC_POLYLENGTH_7B  | Resort to a 7-bit long generating polynomial  |

***CRC polynomial possible sizes actual definitions***

|                    |                 |
|--------------------|-----------------|
| HAL_CRC_LENGTH_32B | 32-bit long CRC |
| HAL_CRC_LENGTH_16B | 16-bit long CRC |
| HAL_CRC_LENGTH_8B  | 8-bit long CRC  |
| HAL_CRC_LENGTH_7B  | 7-bit long CRC  |

## 14 HAL CRC Extension Driver

### 14.1 CRCEEx Firmware driver API description

#### 14.1.1 How to use this driver

- Set user-defined generating polynomial thru HAL\_CRCEEx\_Polynomial\_Set()
- Configure Input or Output data inversion

#### 14.1.2 Detailed description of functions

##### **HAL\_CRCEEx\_Polynomial\_Set**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CRCEEx_Polynomial_Set<br/>(CRC_HandleTypeDef * hcrc, uint32_t Pol, uint32_t PolyLength)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description | Initialize the CRC polynomial if different from default one.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcrc:</b> CRC handle</li> <li>• <b>Pol:</b> CRC generating polynomial (7, 8, 16 or 32-bit long). This parameter is written in normal representation, e.g. <ul style="list-style-type: none"> <li>– for a polynomial of degree 7, <math>X^7 + X^6 + X^5 + X^2 + 1</math> is written 0x65</li> <li>– for a polynomial of degree 16, <math>X^{16} + X^{12} + X^5 + 1</math> is written 0x1021</li> </ul> </li> <li>• <b>PolyLength:</b> CRC polynomial length. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– CRC_POLYLENGTH_7B: 7-bit long CRC (generating polynomial of degree 7)</li> <li>– CRC_POLYLENGTH_8B: 8-bit long CRC (generating polynomial of degree 8)</li> <li>– CRC_POLYLENGTH_16B: 16-bit long CRC (generating polynomial of degree 16)</li> <li>– CRC_POLYLENGTH_32B: 32-bit long CRC (generating polynomial of degree 32)</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

##### **HAL\_CRCEEx\_Input\_Data\_Reverse**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_CRCEEx_Input_Data_Reverse<br/>(CRC_HandleTypeDef * hcrc, uint32_t InputReverseMode)</b>                                                                                                                                                                                                                                                                                                                  |
| Function description | Set the Reverse Input data mode.                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcrc:</b> CRC handle</li> <li>• <b>InputReverseMode:</b> Input Data inversion mode. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– CRC_INPUTDATA_NOINVERSION: no change in bit order (default value)</li> <li>– CRC_INPUTDATA_INVERSION_BYTE: Byte-wise bit reversal</li> <li>– CRC_INPUTDATA_INVERSION_HALFWORD:</li> </ul> </li> </ul> |

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                      | <p>HalfWord-wise bit reversal</p> <ul style="list-style-type: none"> <li>- CRC_INPUTDATA_INVERSION_WORD: Word-wise bit reversal</li> </ul>                                                                                                                                                                                                                                                                                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                            |
|                      | <b>HAL_CRCEx_Output_Data_Reverse</b>                                                                                                                                                                                                                                                                                                                                                                                              |
| Function name        | <b>HAL_StatusTypeDef HAL_CRCEx_Output_Data_Reverse<br/>(CRC_HandleTypeDef * hcrc, uint32_t OutputReverseMode)</b>                                                                                                                                                                                                                                                                                                                 |
| Function description | Set the Reverse Output data mode.                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hcrc:</b> CRC handle</li> <li>• <b>OutputReverseMode:</b> Output Data inversion mode. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- CRC_OUTPUTDATA_INVERSION_DISABLE: no CRC inversion (default value)</li> <li>- CRC_OUTPUTDATA_INVERSION_ENABLE: bit-level inversion (e.g. for a 8-bit CRC: 0xB5 becomes 0xAD)</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                            |

## 14.2 CRCEEx Firmware driver defines

### 14.2.1 CRCEEx

#### *CRC Extended Exported Macros*

|                                         |                                                                                   |
|-----------------------------------------|-----------------------------------------------------------------------------------|
| <u>__HAL_CRC_OUTPUTREVERSAL_ENABLE</u>  | <b>Description:</b>                                                               |
|                                         | <ul style="list-style-type: none"> <li>• Set CRC output reversal.</li> </ul>      |
|                                         | <b>Parameters:</b>                                                                |
|                                         | <ul style="list-style-type: none"> <li>• <u>__HANDLE__</u>: CRC handle</li> </ul> |
|                                         | <b>Return value:</b>                                                              |
|                                         | <ul style="list-style-type: none"> <li>• None.</li> </ul>                         |
| <u>__HAL_CRC_OUTPUTREVERSAL_DISABLE</u> | <b>Description:</b>                                                               |
|                                         | <ul style="list-style-type: none"> <li>• Unset CRC output reversal.</li> </ul>    |
|                                         | <b>Parameters:</b>                                                                |
|                                         | <ul style="list-style-type: none"> <li>• <u>__HANDLE__</u>: CRC handle</li> </ul> |
|                                         | <b>Return value:</b>                                                              |
|                                         | <ul style="list-style-type: none"> <li>• None.</li> </ul>                         |

[\\_\\_HAL\\_CRC\\_POLYNOMIAL\\_CONFIG](#)**Description:**

- Set CRC non-default polynomial.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): CRC handle
- [\\_\\_POLYNOMIAL\\_\\_](#): 7, 8, 16 or 32-bit polynomial

**Return value:**

- None.

***CRC Extended Input Data Inversion Modes***

|                                                         |                                    |
|---------------------------------------------------------|------------------------------------|
| <u><a href="#">CRC_INPUTDATA_INVERSION_NONE</a></u>     | No input data inversion            |
| <u><a href="#">CRC_INPUTDATA_INVERSION_BYTE</a></u>     | Byte-wise input data inversion     |
| <u><a href="#">CRC_INPUTDATA_INVERSION_HALFWORD</a></u> | HalfWord-wise input data inversion |
| <u><a href="#">CRC_INPUTDATA_INVERSION_WORD</a></u>     | Word-wise input data inversion     |

***CRC Extended Output Data Inversion Modes***

|                                                         |                                |
|---------------------------------------------------------|--------------------------------|
| <u><a href="#">CRC_OUTPUTDATA_INVERSION_DISABLE</a></u> | No output data inversion       |
| <u><a href="#">CRC_OUTPUTDATA_INVERSION_ENABLE</a></u>  | Bit-wise output data inversion |

## 15 HAL DAC Generic Driver

### 15.1 DAC Firmware driver registers structures

#### 15.1.1 DAC\_ChannelConfTypeDef

##### Data Fields

- *uint32\_t DAC\_Trigger*
- *uint32\_t DAC\_OutputBuffer*
- *uint32\_t DAC\_OutputSwitch*

##### Field Documentation

- ***uint32\_t DAC\_ChannelConfTypeDef::DAC\_Trigger***  
Specifies the external trigger for the selected DAC channel. This parameter can be a value of [DACEx\\_trigger\\_selection](#)
- ***uint32\_t DAC\_ChannelConfTypeDef::DAC\_OutputBuffer***  
Specifies whether the DAC channel output buffer is enabled or disabled. This parameter can be a value of [DAC\\_output\\_buffer](#) For a given DAC channel, if this paramater applies then DAC\_OutputSwitch does not apply
- ***uint32\_t DAC\_ChannelConfTypeDef::DAC\_OutputSwitch***  
Specifies whether the DAC channel output switch is enabled or disabled. This parameter can be a value of [DAC\\_OutputSwitch](#) For a given DAC channel, if this paramater applies then DAC\_OutputBuffer does not apply

#### 15.1.2 \_\_DAC\_HandleTypeDef

##### Data Fields

- *DAC\_TypeDef \* Instance*
- *\_IO HAL\_DAC\_StateTypeDef State*
- *HAL\_LockTypeDef Lock*
- *DMA\_HandleTypeDef \* DMA\_Handle1*
- *DMA\_HandleTypeDef \* DMA\_Handle2*
- *\_IO uint32\_t ErrorCode*

##### Field Documentation

- ***DAC\_TypeDef\* \_\_DAC\_HandleTypeDef::Instance***  
Register base address
- ***\_IO HAL\_DAC\_StateTypeDef \_\_DAC\_HandleTypeDef::State***  
DAC communication state
- ***HAL\_LockTypeDef \_\_DAC\_HandleTypeDef::Lock***  
DAC locking object
- ***DMA\_HandleTypeDef\* \_\_DAC\_HandleTypeDef::DMA\_Handle1***  
Pointer DMA handler for channel 1U
- ***DMA\_HandleTypeDef\* \_\_DAC\_HandleTypeDef::DMA\_Handle2***  
Pointer DMA handler for channel 2U
- ***\_IO uint32\_t \_\_DAC\_HandleTypeDef::ErrorCode***  
DAC Error code

## 15.2 DAC Firmware driver API description

### 15.2.1 DAC Peripheral features

#### DAC Channels

The device integrates up to 3 12-bit Digital Analog Converters that can be used independently or simultaneously (dual mode):

1. DAC1 channel1 with DAC1\_OUT1 (PA4) as output
2. DAC1 channel2 with DAC1\_OUT2 (PA5) as output (for STM32F3 devices having 2 channels on DAC1)
3. DAC2 channel1 with DAC2\_OUT1 (PA6) as output (for STM32F3 devices having 2 DAC)

#### DAC Triggers

Digital to Analog conversion can be non-triggered using DAC\_TRIGGER\_NONE and DAC1\_OUT1/DAC1\_OUT2/DAC2\_OUT1 is available once writing to DHRx register.

Digital to Analog conversion can be triggered by:

1. External event: EXTI Line 9 (any GPIOx\_PIN\_9) using DAC\_TRIGGER\_EXT\_IT9. The used pin (GPIOx\_PIN\_9) must be configured in input mode.
2. Timers TRGO: TIM2, TIM4, TIM5, TIM6, TIM7 and TIM8 (DAC\_TRIGGER\_T2\_TRGO, DAC\_TRIGGER\_T4\_TRGO...)
3. Software using DAC\_TRIGGER\_SOFTWARE

#### DAC Buffer mode feature

Each DAC channel integrates an output buffer that can be used to reduce the output impedance, and to drive external loads directly without having to add an external operational amplifier. To enable, the output buffer use sConfig.DAC\_OutputBuffer = DAC\_OUTPUTBUFFER\_ENABLE; Or An output switch (in STM32F303x4, STM32F303x6, STM32F303x8 c, STM32F334x6, STM32F334x8 & STM32F334xx). To enable, the output switch sConfig.DAC\_OutputSwitch = DAC\_OUTPUTSWITCH\_ENABLE;



Refer to the device datasheet for more details about output impedance value with and without output buffer.

#### GPIO configurations guidelines

When a DAC channel is used (ex channel1 on PA4) and the other is not (ex channel2 on PA5 is configured in Analog and disabled). Channel1 may disturb channel2 as coupling effect. Note that there is no coupling on channel2 as soon as channel2 is turned on. Coupling on adjacent channel could be avoided as follows: when unused PA5 is configured as INPUT PULL-UP or DOWN. PA5 is configured in ANALOG just before it is turned on.

#### DAC wave generation feature

Both DAC channels of DAC1 can be used to generate note that wave generation is not available in DAC2.

1. Noise wave
2. Triangle wave Wave generation is NOT available in DAC2.

### DAC data format

The DAC data format can be:

1. 8-bit right alignment using DAC\_ALIGN\_8B\_R
2. 12-bit left alignment using DAC\_ALIGN\_12B\_L
3. 12-bit right alignment using DAC\_ALIGN\_12B\_R

### DAC data value to voltage correspondance

The analog output voltage on each DAC channel pin is determined by the following equation:

$$\text{DAC\_OUTx} = \text{VREF+} * \text{DOR} / 4095$$

- with DOR is the Data Output Register

VREF+ is the input voltage reference (refer to the device datasheet)

e.g. To set DAC\_OUT1 to 0.7V, use

- Assuming that VREF+ = 3.3V, DAC\_OUT1 =  $(3.3U * 868U) / 4095U = 0.7V$

### DMA requests

A DMA1 or DMA2 request can be generated when an external trigger (but not a software trigger) occurs if DMA1 or DMA2 requests are enabled using HAL\_DAC\_Start\_DMA().



For Dual mode and specific signal (Triangle and noise) generation please refer to Extended Features Driver description

## 15.2.2 How to use this driver

- DAC APB clock must be enabled to get write access to DAC registers using HAL\_DAC\_Init()
- Configure DAC\_OUTx (DAC\_OUT1: PA4, DAC\_OUT2: PA5) in analog mode.
- Configure the DAC channel using HAL\_DAC\_ConfigChannel() function.
- Enable the DAC channel using HAL\_DAC\_Start() or HAL\_DAC\_Start\_DMA() functions

### Polling mode IO operation

- Start the DAC peripheral using HAL\_DAC\_Start()
- To read the DAC last data output value, use the HAL\_DAC\_GetValue() function.
- Stop the DAC peripheral using HAL\_DAC\_Stop()

### DMA mode IO operation

- Start the DAC peripheral using HAL\_DAC\_Start\_DMA(), at this stage the user specify the length of data to be transferred at each end of conversion
- At the middle of data transfer HAL\_DAC\_ConvHalfCpltCallbackCh1() or HAL\_DACEx\_ConvHalfCpltCallbackCh2() function is executed and user can add his own code by customization of function pointer HAL\_DAC\_ConvHalfCpltCallbackCh1() or HAL\_DACEx\_ConvHalfCpltCallbackCh2()
- At The end of data transfer HAL\_DAC\_ConvCpltCallbackCh1() or HAL\_DACEx\_ConvHalfCpltCallbackCh2() function is executed and user can add his

- own code by customization of function pointer HAL\_DAC\_ConvCpltCallbackCh1() or HAL\_DACEx\_ConvHalfCpltCallbackCh2()
- In case of transfer Error, HAL\_DAC\_ErrorCallbackCh1() function is executed and user can add his own code by customization of function pointer HAL\_DAC\_ErrorCallbackCh1
- In case of DMA underrun, DAC interruption triggers and execute internal function HAL\_DAC\_IRQHandler. HAL\_DAC\_DMAUnderrunCallbackCh1() or HAL\_DACEx\_DMAUnderrunCallbackCh2() function is executed and user can add his own code by customization of function pointer HAL\_DAC\_DMAUnderrunCallbackCh1() or HAL\_DACEx\_DMAUnderrunCallbackCh2() and add his own code by customization of function pointer HAL\_DAC\_ErrorCallbackCh1()
- Stop the DAC peripheral using HAL\_DAC\_Stop\_DMA()

### DAC HAL driver macros list

Below the list of most used macros in DAC HAL driver.

- `__HAL_DAC_ENABLE` : Enable the DAC peripheral
- `__HAL_DAC_DISABLE` : Disable the DAC peripheral
- `__HAL_DAC_CLEAR_FLAG`: Clear the DAC's pending flags
- `__HAL_DAC_GET_FLAG`: Get the selected DAC's flag status



You can refer to the DAC HAL driver header file for more useful macros

### 15.2.3 Initialization and de-initialization functions

This section provides functions allowing to:

- Initialize and configure the DAC.
- De-initialize the DAC.

This section contains the following APIs:

- `HAL_DAC_Init()`
- `HAL_DAC_DeInit()`
- `HAL_DAC_MspInit()`
- `HAL_DAC_MspDeInit()`

### 15.2.4 IO operation functions

This section provides functions allowing to:

- Start conversion.
- Stop conversion.
- Start conversion and enable DMA transfer.
- Stop conversion and disable DMA transfer.
- Get result of conversion.
- Get result of dual mode conversion.

This section contains the following APIs:

- `HAL_DAC_Start()`
- `HAL_DAC_Stop()`
- `HAL_DAC_Stop_DMA()`

- [`HAL\_DAC\_GetValue\(\)`](#)
- [`HAL\_DACEx\_DualGetValue\(\)`](#)
- [`HAL\_DAC\_ConvCpltCallbackCh1\(\)`](#)
- [`HAL\_DAC\_ConvHalfCpltCallbackCh1\(\)`](#)
- [`HAL\_DAC\_ErrorCallbackCh1\(\)`](#)
- [`HAL\_DAC\_DMAUnderrunCallbackCh1\(\)`](#)
- [`HAL\_DAC\_Start\_DMA\(\)`](#)
- [`HAL\_DAC\_ConfigChannel\(\)`](#)
- [`HAL\_DAC\_IRQHandler\(\)`](#)

### 15.2.5 Peripheral Control functions

This section provides functions allowing to:

- Configure channels.
- Configure Triangle wave generation.
- Configure Noise wave generation.
- Set the specified data holding register value for DAC channel.
- Set the specified data holding register value for Dual DAC channels.

This section contains the following APIs:

- [`HAL\_DAC\_ConfigChannel\(\)`](#)
- [`HAL\_DAC\_SetValue\(\)`](#)
- [`HAL\_DACEx\_DualSetValue\(\)`](#)

### 15.2.6 DAC Peripheral State and Error functions

This subsection provides functions allowing to

- Check the DAC state.
- Check the DAC Errors.

This section contains the following APIs:

- [`HAL\_DAC\_GetState\(\)`](#)
- [`HAL\_DAC\_GetError\(\)`](#)

### 15.2.7 Detailed description of functions

#### `HAL_DAC_Init`

|                      |                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_DAC_Init (DAC_HandleTypeDef * hdac)</code>                                                                                                                    |
| Function description | Initialize the DAC peripheral according to the specified parameters in the <code>DAC_InitStruct</code> and initialize the associated handle.                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a <code>DAC_HandleTypeDef</code> structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                    |

**HAL\_DAC\_DeInit**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DAC_DeInit (DAC_HandleTypeDef * hdac)</b>                                                                                                           |
| Function description | Deinitialize the DAC peripheral registers to their default reset values.                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                       |

**HAL\_DAC\_MspInit**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_DAC_MspInit (DAC_HandleTypeDef * hdac)</b>                                                                                                                       |
| Function description | Initialize the DAC MSP.                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_DAC\_MspDeInit**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_DAC_MspDeInit (DAC_HandleTypeDef * hdac)</b>                                                                                                                     |
| Function description | Deinitialize the DAC MSP.                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_DAC\_Start**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DAC_Start (DAC_HandleTypeDef * hdac, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Enables DAC and starts conversion of channel.                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> <li>• <b>Channel:</b> The selected DAC channel. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– DAC_CHANNEL_1: DAC1 Channel1 selected</li> <li>– DAC_CHANNEL_2: DAC1 Channel2 selected</li> <li>– DAC_CHANNEL_1: DAC2 Channel1 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                            |

**HAL\_DAC\_Stop**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DAC_Stop (DAC_HandleTypeDef * hdac, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Disables DAC and stop conversion of channel.                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> <li>• <b>Channel:</b> The selected DAC channel. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– DAC_CHANNEL_1: DAC1 Channel1 selected</li> <li>– DAC_CHANNEL_2: DAC1 Channel2 selected</li> <li>– DAC_CHANNEL_1: DAC2 Channel1 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                            |

**HAL\_DAC\_Start\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DAC_Start_DMA (DAC_HandleTypeDef * hdac, uint32_t Channel, uint32_t * pData, uint32_t Length, uint32_t Alignment)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description | Enables DAC and starts conversion of channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> <li>• <b>Channel:</b> The selected DAC channel. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– DAC_CHANNEL_1: DAC1 Channel1 selected</li> <li>– DAC_CHANNEL_2: DAC1 Channel2 selected</li> </ul> </li> <li>• <b>pData:</b> The destination peripheral Buffer address.</li> <li>• <b>Length:</b> The length of data to be transferred from memory to DAC peripheral</li> <li>• <b>Alignment:</b> Specifies the data alignment for DAC channel. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– DAC_ALIGN_8B_R: 8bit right data alignment selected</li> <li>– DAC_ALIGN_12B_L: 12bit left data alignment selected</li> <li>– DAC_ALIGN_12B_R: 12bit right data alignment selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

**HAL\_DAC\_Stop\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DAC_Stop_DMA (DAC_HandleTypeDef * hdac, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Disables DAC and stop conversion of channel.                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> <li>• <b>Channel:</b> The selected DAC channel. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– DAC_CHANNEL_1: DAC1 Channel1 selected</li> <li>– DAC_CHANNEL_2: DAC1 Channel2 selected</li> <li>– DAC_CHANNEL_1: DAC2 Channel1 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                            |

**HAL\_DAC\_GetValue**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_DAC_GetValue (DAC_HandleTypeDef * hdac,<br/>uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Returns the last data output value of the selected DAC channel.                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> <li>• <b>Channel:</b> The selected DAC channel. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– DAC_CHANNEL_1: DAC1 Channel1 selected</li> <li>– DAC_CHANNEL_2: DAC1 Channel2 selected</li> <li>– DAC_CHANNEL_1: DAC2 Channel1 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>The:</b> selected DAC channel data output value.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                           |

**HAL\_DAC\_ConfigChannel**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_DAC_ConfigChannel<br/>(DAC_HandleTypeDef * hdac, DAC_ChannelConfTypeDef *<br/>sConfig, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Configures the selected DAC channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> <li>• <b>sConfig:</b> DAC configuration structure.</li> <li>• <b>Channel:</b> The selected DAC channel. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– DAC_CHANNEL_1: DAC1 Channel1 selected</li> <li>– DAC_CHANNEL_2: DAC1 Channel2 selected</li> <li>– DAC_CHANNEL_1: DAC2 Channel1 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

**HAL\_DAC\_IRQHandler**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void HAL_DAC_IRQHandler (DAC_HandleTypeDef * hdac)</code>                                                                                                              |
| Function description | Handles DAC interrupt request This function uses the interruption of DMA underrun.                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_DAC\_ConvCpltCallbackCh1**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void HAL_DAC_ConvCpltCallbackCh1 (DAC_HandleTypeDef *<br/>hdac)</code>                                                                                                 |
| Function description | Conversion complete callback in non blocking mode for Channel1.                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_DAC\_ConvHalfCpltCallbackCh1**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_DAC_ConvHalfCpltCallbackCh1<br/>(DAC_HandleTypeDef * hdac)</b>                                                                                                   |
| Function description | Conversion half DMA transfer callback in non blocking mode for Channel1.                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_DAC\_ErrorCallbackCh1**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_DAC_ErrorCallbackCh1 (DAC_HandleTypeDef * hdac)</b>                                                                                                              |
| Function description | Error DAC callback for Channel1.                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_DAC\_DMAUnderrunCallbackCh1**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_DAC_DMAUnderrunCallbackCh1<br/>(DAC_HandleTypeDef * hdac)</b>                                                                                                    |
| Function description | DMA underrun DAC callback for Channel1.                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_DAC\_SetValue**

|                      |                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DAC_SetValue<br/>(DAC_HandleTypeDef * hdac, uint32_t Channel, uint32_t Alignment, uint32_t Data)</b> |
| Function description |                                                                                                                               |

**HAL\_DAC\_GetState**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_DAC_StateTypeDef HAL_DAC_GetState<br/>(DAC_HandleTypeDef * hdac)</b>                                                                                                  |
| Function description | return the DAC handle state                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                        |

**HAL\_DAC\_GetError**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_DAC_GetError (DAC_HandleTypeDef * hdac)</code>                                                                                                            |
| Function description | Return the DAC error code.                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>DAC:</b> Error Code</li> </ul>                                                                                                   |

## 15.3 DAC Firmware driver defines

### 15.3.1 DAC

***DAC data alignment***`DAC_ALIGN_12B_R``DAC_ALIGN_12B_L``DAC_ALIGN_8B_R`***DAC Error Code***`HAL_DAC_ERROR_NONE` No error`HAL_DAC_ERROR_DMAUNDERRUNCH1` DAC channel1 DMA underrun error`HAL_DAC_ERROR_DMAUNDERRUNCH2` DAC channel2 DMA underrun error`HAL_DAC_ERROR_DMA` DMA error***DAC Exported Macros***`_HAL_DAC_RESET_HANDLE_STATE` **Description:**

- Reset DAC handle state.

**Parameters:**

- `_HANDLE_`: specifies the DAC handle.

**Return value:**

- None

`_HAL_DAC_ENABLE`**Description:**

- Enable the DAC channel.

**Parameters:**

- `_HANDLE_`: specifies the DAC handle.
- `_DAC_Channel_`: specifies the DAC channel

**Return value:**

- None

`_HAL_DAC_DISABLE`**Description:**

- Disable the DAC channel.

**Parameters:**

- `_HANDLE_`: specifies the DAC handle

- DAC\_Channel: specifies the DAC channel.

**Return value:**

- None

DAC\_DHR12R1\_ALIGNMENT

**Description:**

- Set DHR12R1 alignment.

**Parameters:**

- ALIGNMENT: specifies the DAC alignment

**Return value:**

- None

DAC\_DHR12R2\_ALIGNMENT

**Description:**

- Set DHR12R2 alignment.

**Parameters:**

- ALIGNMENT: specifies the DAC alignment

**Return value:**

- None

DAC\_DHR12RD\_ALIGNMENT

**Description:**

- Set DHR12RD alignment.

**Parameters:**

- ALIGNMENT: specifies the DAC alignment

**Return value:**

- None

\_HAL\_DAC\_ENABLE\_IT**Description:**

- Enable the DAC interrupt.

**Parameters:**

- HANDLE: specifies the DAC handle
- INTERRUPT: specifies the DAC interrupt. This parameter can be any combination of the following values:
  - DAC\_IT\_DMAUDR1: DAC channel 1 DMA underrun interrupt
  - DAC\_IT\_DMAUDR2: DAC channel 2 DMA underrun interrupt

**Return value:**

- None

\_\_HAL\_DAC\_DISABLE\_IT**Description:**

- Disable the DAC interrupt.

**Parameters:**

- \_\_HANDLE\_\_: specifies the DAC handle
- \_\_INTERRUPT\_\_: specifies the DAC interrupt. This parameter can be any combination of the following values:
  - DAC\_IT\_DMAUDR1: DAC channel 1 DMA underrun interrupt
  - DAC\_IT\_DMAUDR2: DAC channel 2 DMA underrun interrupt

**Return value:**

- None

\_\_HAL\_DAC\_GET\_IT\_SOURCE**Description:**

- Check whether the specified DAC interrupt source is enabled or not.

**Parameters:**

- \_\_HANDLE\_\_: DAC handle
- \_\_INTERRUPT\_\_: DAC interrupt source to check. This parameter can be any combination of the following values:
  - DAC\_IT\_DMAUDR1: DAC channel 1 DMA underrun interrupt
  - DAC\_IT\_DMAUDR2: DAC channel 2 DMA underrun interrupt

**Return value:**

- State: of interruption (SET or RESET)

\_\_HAL\_DAC\_GET\_FLAG**Description:**

- Get the selected DAC's flag status.

**Parameters:**

- \_\_HANDLE\_\_: specifies the DAC handle.
- \_\_FLAG\_\_: specifies the DAC flag to get. This parameter can be any combination of the following values:
  - DAC\_FLAG\_DMAUDR1: DAC channel 1 DMA underrun flag
  - DAC\_FLAG\_DMAUDR2: DAC channel 2 DMA underrun flag

**Return value:**

- None

|                                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>__HAL_DAC_CLEAR_FLAG</code>          | <p><b>Description:</b></p> <ul style="list-style-type: none"> <li>Clear the DAC's flag.</li> </ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"> <li><code>__HANDLE__</code>: specifies the DAC handle.</li> <li><code>__FLAG__</code>: specifies the DAC flag to clear. This parameter can be any combination of the following values: <ul style="list-style-type: none"> <li><code>DAC_FLAG_DMAUDR1</code>: DAC channel 1 DMA underrun flag</li> <li><code>DAC_FLAG_DMAUDR2</code>: DAC channel 2 DMA underrun flag</li> </ul> </li> </ul> <p><b>Return value:</b></p> <ul style="list-style-type: none"> <li>None</li> </ul> |
| <b>DAC flags definition</b>                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <code>DAC_FLAG_DMAUDR1</code>              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <code>DAC_FLAG_DMAUDR2</code>              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>DAC interrupts definition</b>           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <code>DAC_IT_DMAUDR1</code>                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <code>DAC_IT_DMAUDR2</code>                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>DAC Ifsr runmask triangle amplitude</b> |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <code>DAC_LFSRUNMASK_BIT0</code>           | Unmask DAC channel LFSR bit0 for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <code>DAC_LFSRUNMASK_BITS1_0</code>        | Unmask DAC channel LFSR bit[1:0] for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>DAC_LFSRUNMASK_BITS2_0</code>        | Unmask DAC channel LFSR bit[2:0] for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>DAC_LFSRUNMASK_BITS3_0</code>        | Unmask DAC channel LFSR bit[3:0] for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>DAC_LFSRUNMASK_BITS4_0</code>        | Unmask DAC channel LFSR bit[4:0] for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>DAC_LFSRUNMASK_BITS5_0</code>        | Unmask DAC channel LFSR bit[5:0] for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>DAC_LFSRUNMASK_BITS6_0</code>        | Unmask DAC channel LFSR bit[6:0] for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>DAC_LFSRUNMASK_BITS7_0</code>        | Unmask DAC channel LFSR bit[7:0] for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>DAC_LFSRUNMASK_BITS8_0</code>        | Unmask DAC channel LFSR bit[8:0] for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>DAC_LFSRUNMASK_BITS9_0</code>        | Unmask DAC channel LFSR bit[9:0] for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>DAC_LFSRUNMASK_BITS10_0</code>       | Unmask DAC channel LFSR bit[10:0] for noise wave generation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

|                            |                                                             |
|----------------------------|-------------------------------------------------------------|
| DAC_LFSRUNMASK_BITS11_0    | Unmask DAC channel LFSR bit[11:0] for noise wave generation |
| DAC_TRIANGLEAMPLITUDE_1    | Select max triangle amplitude of 1U                         |
| DAC_TRIANGLEAMPLITUDE_3    | Select max triangle amplitude of 3U                         |
| DAC_TRIANGLEAMPLITUDE_7    | Select max triangle amplitude of 7U                         |
| DAC_TRIANGLEAMPLITUDE_15   | Select max triangle amplitude of 15U                        |
| DAC_TRIANGLEAMPLITUDE_31   | Select max triangle amplitude of 31U                        |
| DAC_TRIANGLEAMPLITUDE_63   | Select max triangle amplitude of 63U                        |
| DAC_TRIANGLEAMPLITUDE_127  | Select max triangle amplitude of 127U                       |
| DAC_TRIANGLEAMPLITUDE_255  | Select max triangle amplitude of 255U                       |
| DAC_TRIANGLEAMPLITUDE_511  | Select max triangle amplitude of 511U                       |
| DAC_TRIANGLEAMPLITUDE_1023 | Select max triangle amplitude of 1023U                      |
| DAC_TRIANGLEAMPLITUDE_2047 | Select max triangle amplitude of 2047U                      |
| DAC_TRIANGLEAMPLITUDE_4095 | Select max triangle amplitude of 4095U                      |
| <b>DAC output buffer</b>   |                                                             |
| DAC_OUTPUTBUFFER_ENABLE    |                                                             |
| DAC_OUTPUTBUFFER_DISABLE   |                                                             |

## 16 HAL DAC Extension Driver

### 16.1 DACEx Firmware driver API description

#### 16.1.1 How to use this driver

- When Dual mode is enabled (i.e. DAC Channel1 and Channel2 are used simultaneously) : Use HAL\_DACEx\_DualGetValue() to get digital data to be converted and use HAL\_DACEx\_DualSetValue() to set digital value to converted simultaneously in Channel 1 and Channel 2.
- Use HAL\_DACEx\_TriangleWaveGenerate() to generate Triangle signal.
- Use HAL\_DACEx\_NoiseWaveGenerate() to generate Noise signal.

#### 16.1.2 Peripheral Control functions

This section provides functions allowing to:

- Set the specified data holding register value for DAC channel.
- Set the specified data holding register value for dual DAC channel (when DAC channel 2 is present in DAC 1U)

This section contains the following APIs:

- [`HAL\_DAC\_SetValue\(\)`](#)
- [`HAL\_DACEx\_DualSetValue\(\)`](#)

#### 16.1.3 IO operation functions

This section provides functions allowing to:

- Start conversion.
- Start conversion and enable DMA transfer.
- Get result of conversion.
- Handle DAC IRQ's.
- Generate triangular-wave
- Generate noise-wave
- Callback functions for DAC1 Channel2 (when supported)

This section contains the following APIs:

- [`HAL\_DAC\_Start\(\)`](#)
- [`HAL\_DAC\_Start\_DMA\(\)`](#)
- [`HAL\_DAC\_GetValue\(\)`](#)
- [`HAL\_DACEx\_DualGetValue\(\)`](#)
- [`HAL\_DAC\_IRQHandler\(\)`](#)
- [`HAL\_DAC\_ConfigChannel\(\)`](#)
- [`HAL\_DACEx\_TriangleWaveGenerate\(\)`](#)
- [`HAL\_DACEx\_NoiseWaveGenerate\(\)`](#)
- [`HAL\_DACEx\_ConvCpltCallbackCh2\(\)`](#)
- [`HAL\_DACEx\_ConvHalfCpltCallbackCh2\(\)`](#)
- [`HAL\_DACEx\_ErrorCallbackCh2\(\)`](#)
- [`HAL\_DACEx\_DMAUnderrunCallbackCh2\(\)`](#)
- [`HAL\_DACEx\_DualSetValue\(\)`](#)

### 16.1.4 Detailed description of functions

#### HAL\_DACEx\_DualGetValue

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_DACEx_DualGetValue (DAC_HandleTypeDef * hdac)</code>                                                                                                      |
| Function description | Return the last data output value of the selected DAC channel.                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>The:</b> selected DAC channel data output value.</li> </ul>                                                                      |

#### HAL\_DACEx\_DualSetValue

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_DACEx_DualSetValue (DAC_HandleTypeDef * hdac, uint32_t Alignment, uint32_t Data1, uint32_t Data2)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description | Set the specified data holding register value for dual DAC channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> <li>• <b>Alignment:</b> Specifies the data alignment for dual channel DAC. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– DAC_ALIGN_8B_R: 8bit right data alignment selected</li> <li>– DAC_ALIGN_12B_L: 12bit left data alignment selected</li> <li>– DAC_ALIGN_12B_R: 12bit right data alignment selected</li> </ul> </li> <li>• <b>Data2:</b> Data for DAC Channel2 to be loaded in the selected data holding register.</li> <li>• <b>Data1:</b> Data for DAC Channel1 to be loaded in the selected data holding register.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• In dual mode, a unique register access is required to write in both DAC channels at the same time.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

#### HAL\_DACEx\_TriangleWaveGenerate

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_DACEx_TriangleWaveGenerate (DAC_HandleTypeDef * hdac, uint32_t Channel, uint32_t Amplitude)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Enables or disables the selected DAC channel wave generation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> <li>• <b>Channel:</b> The selected DAC channel. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– DAC_CHANNEL_1: DAC1 Channel1 selected</li> <li>– DAC_CHANNEL_2: DAC1 Channel2 selected</li> </ul> </li> <li>• <b>Amplitude:</b> Select max triangle amplitude. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– DAC_TRIANGLEAMPLITUDE_1: Select max triangle amplitude of 1</li> <li>– DAC_TRIANGLEAMPLITUDE_3: Select max triangle</li> </ul> </li> </ul> |

- amplitude of 3
- DAC\_TRIANGLEAMPLITUDE\_7: Select max triangle amplitude of 7
- DAC\_TRIANGLEAMPLITUDE\_15: Select max triangle amplitude of 15
- DAC\_TRIANGLEAMPLITUDE\_31: Select max triangle amplitude of 31
- DAC\_TRIANGLEAMPLITUDE\_63: Select max triangle amplitude of 63
- DAC\_TRIANGLEAMPLITUDE\_127: Select max triangle amplitude of 127
- DAC\_TRIANGLEAMPLITUDE\_255: Select max triangle amplitude of 255
- DAC\_TRIANGLEAMPLITUDE\_511: Select max triangle amplitude of 511
- DAC\_TRIANGLEAMPLITUDE\_1023: Select max triangle amplitude of 1023
- DAC\_TRIANGLEAMPLITUDE\_2047: Select max triangle amplitude of 2047
- DAC\_TRIANGLEAMPLITUDE\_4095: Select max triangle amplitude of 4095

Return values

- **HAL:** status

Notes

- Wave generation is not available in DAC2.

### **HAL\_DACEx\_NoiseWaveGenerate**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DACEx_NoiseWaveGenerate<br/>(DAC_HandleTypeDef * hdac, uint32_t Channel, uint32_t Amplitude)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Enables or disables the selected DAC channel wave generation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> <li>• <b>Channel:</b> The selected DAC channel. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- DAC_CHANNEL_1: DAC1 Channel1 selected</li> <li>- DAC_CHANNEL_2: DAC1 Channel2 selected</li> </ul> </li> <li>• <b>Amplitude:</b> Unmask DAC channel LFSR for noise wave generation. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- DAC_LFSRUNMASK_BIT0: Unmask DAC channel LFSR bit0 for noise wave generation</li> <li>- DAC_LFSRUNMASK_BITS1_0: Unmask DAC channel LFSR bit[1:0] for noise wave generation</li> <li>- DAC_LFSRUNMASK_BITS2_0: Unmask DAC channel LFSR bit[2:0] for noise wave generation</li> <li>- DAC_LFSRUNMASK_BITS3_0: Unmask DAC channel LFSR bit[3:0] for noise wave generation</li> <li>- DAC_LFSRUNMASK_BITS4_0: Unmask DAC channel LFSR bit[4:0] for noise wave generation</li> <li>- DAC_LFSRUNMASK_BITS5_0: Unmask DAC channel LFSR bit[5:0] for noise wave generation</li> <li>- DAC_LFSRUNMASK_BITS6_0: Unmask DAC channel</li> </ul> </li> </ul> |



- LFSR bit[6:0] for noise wave generation
- DAC\_LFSRUNMASK\_BITS7\_0: Unmask DAC channel LFSR bit[7:0] for noise wave generation
- DAC\_LFSRUNMASK\_BITS8\_0: Unmask DAC channel LFSR bit[8:0] for noise wave generation
- DAC\_LFSRUNMASK\_BITS9\_0: Unmask DAC channel LFSR bit[9:0] for noise wave generation
- DAC\_LFSRUNMASK\_BITS10\_0: Unmask DAC channel LFSR bit[10:0] for noise wave generation
- DAC\_LFSRUNMASK\_BITS11\_0: Unmask DAC channel LFSR bit[11:0] for noise wave generation

Return values

- **HAL:** status

### **HAL\_DACEx\_ConvCpltCallbackCh2**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_DACEx_ConvCpltCallbackCh2<br/>(DAC_HandleTypeDef * hdac)</b>                                                                                                     |
| Function description | Conversion complete callback in non blocking mode for Channel2.                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

### **HAL\_DACEx\_ConvHalfCpltCallbackCh2**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_DACEx_ConvHalfCpltCallbackCh2<br/>(DAC_HandleTypeDef * hdac)</b>                                                                                                 |
| Function description | Conversion half DMA transfer callback in non blocking mode for Channel2.                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

### **HAL\_DACEx\_ErrorCallbackCh2**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_DACEx_ErrorCallbackCh2 (DAC_HandleTypeDef * hdac)</b>                                                                                                            |
| Function description | Error DAC callback for Channel2.                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_DACEx\_DMAUnderrunCallbackCh2**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_DACEx_DMAUnderrunCallbackCh2<br/>(DAC_HandleTypeDef * hdac)</b>                                                                                                  |
| Function description | DMA underrun DAC callback for channel2.                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdac:</b> pointer to a DAC_HandleTypeDef structure that contains the configuration information for the specified DAC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**16.2 DACEx Firmware driver defines****16.2.1 DACEx*****DACEx Channel selection***

|               |                |
|---------------|----------------|
| DAC_CHANNEL_1 | DAC Channel 1U |
| DAC_CHANNEL_2 | DAC Channel 2U |

***DACEx trigger selection***

|                      |                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------|
| DAC_TRIGGER_NONE     | Conversion is automatic once the DAC1_DHRxxxx register has been loaded, and not by external trigger |
| DAC_TRIGGER_T2_TRGO  | TIM2 TRGO selected as external conversion trigger for DAC channel                                   |
| DAC_TRIGGER_T4_TRGO  | TIM4 TRGO selected as external conversion trigger for DAC channel                                   |
| DAC_TRIGGER_T15_TRGO | TIM5 TRGO selected as external conversion trigger for DAC channel                                   |
| DAC_TRIGGER_T6_TRGO  | TIM6 TRGO selected as external conversion trigger for DAC channel                                   |
| DAC_TRIGGER_T7_TRGO  | TIM7 TRGO selected as external conversion trigger for DAC channel                                   |
| DAC_TRIGGER_T3_TRGO  | TIM3 TRGO selected as external conversion trigger for DAC channel Use                               |
| DAC_TRIGGER_T8_TRGO  | TIM8 TRGO selected as external conversion trigger for DAC channel Use                               |
| DAC_TRIGGER_EXT_IT9  | EXTI Line9 event selected as external conversion trigger for DAC channel                            |
| DAC_TRIGGER_SOFTWARE | Conversion started by software trigger for DAC channel                                              |
| IS_DAC_TRIGGER       |                                                                                                     |

## 17 HAL DMA Generic Driver

### 17.1 DMA Firmware driver registers structures

#### 17.1.1 DMA\_InitTypeDef

##### Data Fields

- *uint32\_t Direction*
- *uint32\_t PeriphInc*
- *uint32\_t MemInc*
- *uint32\_t PeriphDataAlignment*
- *uint32\_t MemDataAlignment*
- *uint32\_t Mode*
- *uint32\_t Priority*

##### Field Documentation

- ***uint32\_t DMA\_InitTypeDef::Direction***  
Specifies if the data will be transferred from memory to peripheral, from memory to memory or from peripheral to memory. This parameter can be a value of [\*\*DMA\\_Data\\_transfer\\_direction\*\*](#)
- ***uint32\_t DMA\_InitTypeDef::PeriphInc***  
Specifies whether the Peripheral address register should be incremented or not. This parameter can be a value of [\*\*DMA\\_Peripheral\\_incremented\\_mode\*\*](#)
- ***uint32\_t DMA\_InitTypeDef::MemInc***  
Specifies whether the memory address register should be incremented or not. This parameter can be a value of [\*\*DMA\\_Memory\\_incremented\\_mode\*\*](#)
- ***uint32\_t DMA\_InitTypeDef::PeriphDataAlignment***  
Specifies the Peripheral data width. This parameter can be a value of [\*\*DMA\\_Peripheral\\_data\\_size\*\*](#)
- ***uint32\_t DMA\_InitTypeDef::MemDataAlignment***  
Specifies the Memory data width. This parameter can be a value of [\*\*DMA\\_Memory\\_data\\_size\*\*](#)
- ***uint32\_t DMA\_InitTypeDef::Mode***  
Specifies the operation mode of the DMAy Channelx. This parameter can be a value of [\*\*DMA\\_mode\*\*](#)  
**Note:**The circular buffer mode cannot be used if the memory-to-memory data transfer is configured on the selected Channel
- ***uint32\_t DMA\_InitTypeDef::Priority***  
Specifies the software priority for the DMAy Channelx. This parameter can be a value of [\*\*DMA\\_Priority\\_level\*\*](#)

#### 17.1.2 \_\_DMA\_HandleTypeDef

##### Data Fields

- *DMA\_Channel\_TypeDef \* Instance*
- *DMA\_InitTypeDef Init*
- *HAL\_LockTypeDef Lock*
- *HAL\_DMA\_StateTypeDef State*
- *void \* Parent*
- *void(\* XferCpltCallback*
- *void(\* XferHalfCpltCallback*

- `void(* XferErrorCallback`
- `void(* XferAbortCallback`
- `__IO uint32_t ErrorCode`
- `DMA_TypeDef * DmaBaseAddress`
- `uint32_t ChannelIndex`

#### Field Documentation

- `DMA_Channel_TypeDef* __DMA_HandleTypeDef::Instance`  
Register base address
- `DMA_InitTypeDef __DMA_HandleTypeDef::Init`  
DMA communication parameters
- `HAL_LockTypeDef __DMA_HandleTypeDef::Lock`  
DMA locking object
- `HAL_DMA_StateTypeDef __DMA_HandleTypeDef::State`  
DMA transfer state
- `void* __DMA_HandleTypeDef::Parent`  
Parent object state
- `void(* __DMA_HandleTypeDef::XferCpltCallback)(struct __DMA_HandleTypeDef *hdma)`  
DMA transfer complete callback
- `void(* __DMA_HandleTypeDef::XferHalfCpltCallback)(struct __DMA_HandleTypeDef *hdma)`  
DMA Half transfer complete callback
- `void(* __DMA_HandleTypeDef::XferErrorCallback)(struct __DMA_HandleTypeDef *hdma)`  
DMA transfer error callback
- `void(* __DMA_HandleTypeDef::XferAbortCallback)(struct __DMA_HandleTypeDef *hdma)`  
DMA transfer abort callback
- `__IO uint32_t __DMA_HandleTypeDef::ErrorCode`  
DMA Error code
- `DMA_TypeDef* __DMA_HandleTypeDef::DmaBaseAddress`  
DMA Channel Base Address
- `uint32_t __DMA_HandleTypeDef::ChannelIndex`  
DMA Channel Index

## 17.2 DMA Firmware driver API description

### 17.2.1 How to use this driver

1. Enable and configure the peripheral to be connected to the DMA Channel (except for internal SRAM / FLASH memories: no initialization is necessary). Please refer to Reference manual for connection between peripherals and DMA requests .
2. For a given Channel, program the required configuration through the following parameters: Transfer Direction, Source and Destination data formats, Circular or Normal mode, Channel Priority level, Source and Destination Increment mode, using HAL\_DMA\_Init() function.
3. Use HAL\_DMA\_GetState() function to return the DMA state and HAL\_DMA\_GetError() in case of error detection.
4. Use HAL\_DMA\_Abort() function to abort the current transfer In Memory-to-Memory transfer mode, Circular mode is not allowed.

### Polling mode IO operation

- Use HAL\_DMA\_Start() to start DMA transfer after the configuration of Source address and destination address and the Length of data to be transferred
- Use HAL\_DMA\_PollForTransfer() to poll for the end of current transfer, in this case a fixed Timeout can be configured by User depending from his application.

### Interrupt mode IO operation

- Configure the DMA interrupt priority using HAL\_NVIC\_SetPriority()
- Enable the DMA IRQ handler using HAL\_NVIC\_EnableIRQ()
- Use HAL\_DMA\_Start\_IT() to start DMA transfer after the configuration of Source address and destination address and the Length of data to be transferred. In this case the DMA interrupt is configured
- Use HAL\_DMA\_Channel\_IRQHandler() called under DMA\_IRQHandler() Interrupt subroutine
- At the end of data transfer HAL\_DMA\_IRQHandler() function is executed and user can add his own function by customization of function pointer XferCpltCallback and XferErrorCallback (i.e a member of DMA handle structure).

### DMA HAL driver macros list

Below the list of most used macros in DMA HAL driver.



You can refer to the DMA HAL driver header file for more useful macros

## 17.2.2 Initialization and de-initialization functions

This section provides functions allowing to initialize the DMA Channel source and destination addresses, incrementation and data sizes, transfer direction, circular/normal mode selection, memory-to-memory mode selection and Channel priority value.

The HAL\_DMA\_Init() function follows the DMA configuration procedures as described in reference manual.

This section contains the following APIs:

- [\*\*HAL\\_DMA\\_Init\(\)\*\*](#)
- [\*\*HAL\\_DMA\\_DeInit\(\)\*\*](#)

## 17.2.3 IO operation functions

This section provides functions allowing to:

- Configure the source, destination address and data length and Start DMA transfer
- Configure the source, destination address and data length and Start DMA transfer with interrupt
- Abort DMA transfer
- Poll for transfer complete
- Handle DMA interrupt request

This section contains the following APIs:

- [\*\*HAL\\_DMA\\_Start\(\)\*\*](#)
- [\*\*HAL\\_DMA\\_Start\\_IT\(\)\*\*](#)
- [\*\*HAL\\_DMA\\_Abort\(\)\*\*](#)

- [\*HAL\\_DMA\\_Abort\\_IT\(\)\*](#)
- [\*HAL\\_DMA\\_PollForTransfer\(\)\*](#)
- [\*HAL\\_DMA\\_IRQHandler\(\)\*](#)
- [\*HAL\\_DMA\\_RegisterCallback\(\)\*](#)
- [\*HAL\\_DMA\\_UnRegisterCallback\(\)\*](#)

#### 17.2.4 State and Errors functions

This subsection provides functions allowing to

- Check the DMA state
- Get error code

This section contains the following APIs:

- [\*HAL\\_DMA\\_GetState\(\)\*](#)
- [\*HAL\\_DMA\\_GetError\(\)\*](#)

#### 17.2.5 Detailed description of functions

##### **HAL\_DMA\_Init**

|                      |                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DMA_Init (DMA_HandleTypeDef * hdma)</b>                                                                                                                     |
| Function description | Initialize the DMA according to the specified parameters in the DMA_InitTypeDef and initialize the associated handle.                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> Pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Channel.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                               |

##### **HAL\_DMA\_DeInit**

|                      |                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DMA_DeInit (DMA_HandleTypeDef * hdma)</b>                                                                                                                   |
| Function description | Deinitialize the DMA peripheral.                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Channel.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                               |

##### **HAL\_DMA\_Start**

|                      |                                                                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DMA_Start (DMA_HandleTypeDef * hdma, uint32_t SrcAddress, uint32_t DstAddress, uint32_t DataLength)</b>                                                                                                                                                                                          |
| Function description | Start the DMA Transfer.                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> : pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Channel.</li> <li>• <b>SrcAddress:</b> The source memory Buffer address</li> <li>• <b>DstAddress:</b> The destination memory Buffer address</li> </ul> |

- **DataLength:** The length of data to be transferred from source to destination
- **HAL:** status

### **HAL\_DMA\_Start\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DMA_Start_IT<br/>(DMA_HandleTypeDef * hdma, uint32_t SrcAddress, uint32_t DstAddress, uint32_t DataLength)</b>                                                                                                                                                                                                                                                                               |
| Function description | Start the DMA Transfer with interrupt enabled.                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Channel.</li> <li>• <b>SrcAddress:</b> The source memory Buffer address</li> <li>• <b>DstAddress:</b> The destination memory Buffer address</li> <li>• <b>DataLength:</b> The length of data to be transferred from source to destination</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                  |

### **HAL\_DMA\_Abort**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DMA_Abort (DMA_HandleTypeDef * hdma)</b>                                                                                                                      |
| Function description | Abort the DMA Transfer.                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> : pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Channel.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                   |

### **HAL\_DMA\_Abort\_IT**

|                      |                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DMA_Abort_IT<br/>(DMA_HandleTypeDef * hdma)</b>                                                                                                              |
| Function description | Abort the DMA Transfer in Interrupt mode.                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> : pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Stream.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                  |

### **HAL\_DMA\_PollForTransfer**

|                      |                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DMA_PollForTransfer<br/>(DMA_HandleTypeDef * hdma, uint32_t CompleteLevel,<br/>uint32_t Timeout)</b>                                                        |
| Function description | Polling for transfer complete.                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Channel.</li> </ul> |

---

|               |                                                                                                                                                          |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>• <b>CompleteLevel:</b> Specifies the DMA level complete.</li> <li>• <b>Timeout:</b> Timeout duration.</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                   |

### HAL\_DMA\_IRQHandler

|                      |                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_DMA_IRQHandler (DMA_HandleTypeDef * hdma)</b>                                                                                                                            |
| Function description | Handle DMA interrupt request.                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Channel.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                      |

### HAL\_DMA\_RegisterCallback

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DMA_RegisterCallback<br/>(DMA_HandleTypeDef * hdma, HAL_DMA_CallbackIDTypeDef<br/>CallbackID, void(*)(DMA_HandleTypeDef * _hdma) pCallback)</b>                                                                                                                                                                                                                                                     |
| Function description | Register callbacks.                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Stream.</li> <li>• <b>CallbackID:</b> User Callback identifier a HAL_DMA_CallbackIDTypeDef ENUM as parameter.</li> <li>• <b>pCallback:</b> pointer to private callback function which has pointer to a DMA_HandleTypeDef structure as parameter.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                       |

### HAL\_DMA\_UnRegisterCallback

|                      |                                                                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_DMA_UnRegisterCallback<br/>(DMA_HandleTypeDef * hdma, HAL_DMA_CallbackIDTypeDef<br/>CallbackID)</b>                                                                                                                                                              |
| Function description | UnRegister callbacks.                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Stream.</li> <li>• <b>CallbackID:</b> User Callback identifier a HAL_DMA_CallbackIDTypeDef ENUM as parameter.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                    |

**HAL\_DMA\_GetState**

|                      |                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_DMA_StateTypeDef HAL_DMA_GetState<br/>(DMA_HandleTypeDef * hdma)</b>                                                                                                          |
| Function description | Returns the DMA state.                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Channel.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                                |

**HAL\_DMA\_GetError**

|                      |                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_DMA_GetError (DMA_HandleTypeDef * hdma)</b>                                                                                                                          |
| Function description | Return the DMA error code.                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to a DMA_HandleTypeDef structure that contains the configuration information for the specified DMA Channel.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>DMA:</b> Error Code</li> </ul>                                                                                                           |

## 17.3 DMA Firmware driver defines

### 17.3.1 DMA

*DMA Data transfer direction*

|                             |                                |
|-----------------------------|--------------------------------|
| <b>DMA_PERIPH_TO_MEMORY</b> | Peripheral to memory direction |
| <b>DMA_MEMORY_TO_PERIPH</b> | Memory to peripheral direction |
| <b>DMA_MEMORY_TO_MEMORY</b> | Memory to memory direction     |

*DMA Error Code*

|                                    |                     |
|------------------------------------|---------------------|
| <b>HAL_DMA_ERROR_NONE</b>          | No error            |
| <b>HAL_DMA_ERROR_TE</b>            | Transfer error      |
| <b>HAL_DMA_ERROR_NO_XFER</b>       | no ongoing transfer |
| <b>HAL_DMA_ERROR_TIMEOUT</b>       | Timeout error       |
| <b>HAL_DMA_ERROR_NOT_SUPPORTED</b> | Not supported mode  |

*DMA Exported Macros***\_HAL\_DMA\_RESET\_HANDLE\_STATE** **Description:**

- Reset DMA handle state.

**Parameters:**

- **\_HANDLE\_**: DMA handle.

**Return value:**

- None

`__HAL_DMA_ENABLE`

**Description:**

- Enable the specified DMA Channel.

**Parameters:**

- `__HANDLE__`: DMA handle

**Return value:**

- None

`__HAL_DMA_DISABLE`

**Description:**

- Disable the specified DMA Channel.

**Parameters:**

- `__HANDLE__`: DMA handle

**Return value:**

- None

`__HAL_DMA_ENABLE_IT`

**Description:**

- Enables the specified DMA Channel interrupts.

**Parameters:**

- `__HANDLE__`: DMA handle
- `__INTERRUPT__`: specifies the DMA interrupt sources to be enabled or disabled. This parameter can be any combination of the following values:
  - `DMA_IT_TC`: Transfer complete interrupt mask
  - `DMA_IT_HT`: Half transfer complete interrupt mask
  - `DMA_IT_TE`: Transfer error interrupt mask

**Return value:**

- None

`__HAL_DMA_DISABLE_IT`

**Description:**

- Disables the specified DMA Channel interrupts.

**Parameters:**

- `__HANDLE__`: DMA handle
- `__INTERRUPT__`: specifies the DMA interrupt sources to be enabled or disabled. This parameter can be any combination of the following values:
  - `DMA_IT_TC`: Transfer complete interrupt mask
  - `DMA_IT_HT`: Half transfer complete interrupt mask

- DMA\_IT\_TE: Transfer error interrupt mask

**Return value:**

- None

`__HAL_DMA_GET_IT_SOURCE`

- Description:**

  - Checks whether the specified DMA Channel interrupt is enabled or disabled.

**Parameters:**

- `__HANDLE__`: DMA handle
- `__INTERRUPT__`: specifies the DMA interrupt source to check. This parameter can be one of the following values:
  - DMA\_IT\_TC: Transfer complete interrupt mask
  - DMA\_IT\_HT: Half transfer complete interrupt mask
  - DMA\_IT\_TE: Transfer error interrupt mask

**Return value:**

- The: state of DMA\_IT (SET or RESET).

`__HAL_DMA_GET_COUNTER`

- Description:**

  - Returns the number of remaining data units in the current DMAy Channelx transfer.

**Parameters:**

- `__HANDLE__`: DMA handle

**Return value:**

- The: number of remaining data units in the current DMA Channel transfer.

**DMA flag definitions**

`DMA_FLAG_GL1`  
`DMA_FLAG_TC1`  
`DMA_FLAG_HT1`  
`DMA_FLAG_TE1`  
`DMA_FLAG_GL2`  
`DMA_FLAG_TC2`  
`DMA_FLAG_HT2`  
`DMA_FLAG_TE2`  
`DMA_FLAG_GL3`  
`DMA_FLAG_TC3`  
`DMA_FLAG_HT3`

DMA\_FLAG\_TE3  
DMA\_FLAG\_GL4  
DMA\_FLAG\_TC4  
DMA\_FLAG\_HT4  
DMA\_FLAG\_TE4  
DMA\_FLAG\_GL5  
DMA\_FLAG\_TC5  
DMA\_FLAG\_HT5  
DMA\_FLAG\_TE5  
DMA\_FLAG\_GL6  
DMA\_FLAG\_TC6  
DMA\_FLAG\_HT6  
DMA\_FLAG\_TE6  
DMA\_FLAG\_GL7  
DMA\_FLAG\_TC7  
DMA\_FLAG\_HT7  
DMA\_FLAG\_TE7

***DMA interrupt enable definitions***

DMA\_IT\_TC  
DMA\_IT\_HT  
DMA\_IT\_TE

***DMA Memory data size***

DMA\_MDATAALIGN\_BYTE      Memory data alignment : Byte  
DMA\_MDATAALIGN\_HALFWORD    Memory data alignment : HalfWord  
DMA\_MDATAALIGN\_WORD        Memory data alignment : Word

***DMA Memory incremented mode***

DMA\_MINC\_ENABLE    Memory increment mode Enable  
DMA\_MINC\_DISABLE   Memory increment mode Disable

***DMA mode***

DMA\_NORMAL            Normal Mode  
DMA\_CIRCULAR          Circular Mode

***DMA Peripheral data size***

DMA\_PDATAALIGN\_BYTE    Peripheral data alignment : Byte  
DMA\_PDATAALIGN\_HALFWORD Peripheral data alignment : HalfWord  
DMA\_PDATAALIGN\_WORD     Peripheral data alignment : Word

**DMA Peripheral incremented mode**

`DMA_PINC_ENABLE` Peripheral increment mode Enable

`DMA_PINC_DISABLE` Peripheral increment mode Disable

**DMA Priority level**

`DMA_PRIORITY_LOW` Priority level : Low

`DMA_PRIORITY_MEDIUM` Priority level : Medium

`DMA_PRIORITY_HIGH` Priority level : High

`DMA_PRIORITY VERY_HIGH` Priority level : Very\_High

**DMA Remap Enable**

`__HAL_DMA_REMAP_CHANNEL_ENABLE` **Description:**

- DMA remapping enable/disable macros.

**Parameters:**

- `__DMA_REMAP__`: This parameter can be a value of

`__HAL_DMA_REMAP_CHANNEL_DISABLE`

## 18 HAL DMA Extension Driver

### 18.1 DMAEx Firmware driver defines

#### 18.1.1 DMAEx

##### *DMA Extended Exported Macros*

- `__HAL_DMA_GET_TC_FLAG_INDEX` **Description:**
- Returns the current DMA Channel transfer complete flag.
- Parameters:**
- `__HANDLE__`: DMA handle
- Return value:**
- The: specified transfer complete flag index.
- `__HAL_DMA_GET_HT_FLAG_INDEX` **Description:**
- Returns the current DMA Channel half transfer complete flag.
- Parameters:**
- `__HANDLE__`: DMA handle
- Return value:**
- The: specified half transfer complete flag index.
- `__HAL_DMA_GET_TE_FLAG_INDEX` **Description:**
- Returns the current DMA Channel transfer error flag.
- Parameters:**
- `__HANDLE__`: DMA handle
- Return value:**
- The: specified transfer error flag index.
- `__HAL_DMA_GET_GI_FLAG_INDEX` **Description:**
- Return the current DMA Channel Global interrupt flag.
- Parameters:**
- `__HANDLE__`: DMA handle
- Return value:**
- The: specified transfer error flag index.
- `__HAL_DMA_GET_FLAG` **Description:**
- Get the DMA Channel pending flags.

**Parameters:**

- `_HANDLE_`: DMA handle
- `_FLAG_`: Get the specified flag. This parameter can be any combination of the following values:
  - `DMA_FLAG_TCx`: Transfer complete flag
  - `DMA_FLAG_HTx`: Half transfer complete flag
  - `DMA_FLAG_TEx`: Transfer error flag Where x can be `1_7` or `1_5` (depending on DMA1 or DMA2) to select the DMA Channel flag.

**Return value:**

- The state of FLAG (SET or RESET).

`__HAL_DMA_CLEAR_FLAG`

**Description:**

- Clears the DMA Channel pending flags.

**Parameters:**

- `_HANDLE_`: DMA handle
- `_FLAG_`: specifies the flag to clear. This parameter can be any combination of the following values:
  - `DMA_FLAG_TCx`: Transfer complete flag
  - `DMA_FLAG_HTx`: Half transfer complete flag
  - `DMA_FLAG_TEx`: Transfer error flag Where x can be `1_7` or `1_5` (depending on DMA1 or DMA2) to select the DMA Channel flag.

**Return value:**

- None

## 19 HAL FLASH Generic Driver

### 19.1 FLASH Firmware driver registers structures

#### 19.1.1 FLASH\_ProcTypeDef

##### Data Fields

- `__IO FLASH_ProcTypeDef ProcedureOnGoing`
- `__IO uint32_t DataRemaining`
- `__IO uint32_t Address`
- `__IO uint64_t Data`
- `HAL_LockTypeDef Lock`
- `__IO uint32_t ErrorCode`

##### Field Documentation

- `__IO FLASH_ProcTypeDef FLASH_ProcTypeDef::ProcedureOnGoing`  
Internal variable to indicate which procedure is ongoing or not in IT context
- `__IO uint32_t FLASH_ProcTypeDef::DataRemaining`  
Internal variable to save the remaining pages to erase or half-word to program in IT context
- `__IO uint32_t FLASH_ProcTypeDef::Address`  
Internal variable to save address selected for program or erase
- `__IO uint64_t FLASH_ProcTypeDef::Data`  
Internal variable to save data to be programmed
- `HAL_LockTypeDef FLASH_ProcTypeDef::Lock`  
FLASH locking object
- `__IO uint32_t FLASH_ProcTypeDef::ErrorCode`  
FLASH error code This parameter can be a value of [FLASH\\_Error\\_Codes](#)

### 19.2 FLASH Firmware driver API description

#### 19.2.1 FLASH peripheral features

The Flash memory interface manages CPU AHB I-Code and D-Code accesses to the Flash memory. It implements the erase and program Flash memory operations and the read and write protection mechanisms.

The Flash memory interface accelerates code execution with a system of instruction prefetch.

The FLASH main features are:

- Flash memory read operations
- Flash memory program/erase operations
- Read / write protections
- Prefetch on I-Code
- Option Bytes programming

### 19.2.2 How to use this driver

This driver provides functions and macros to configure and program the FLASH memory of all STM32F3xx devices.

1. FLASH Memory I/O Programming functions: this group includes all needed functions to erase and program the main memory:
  - Lock and Unlock the FLASH interface
  - Erase function: Erase page, erase all pages
  - Program functions: half word, word and doubleword
2. FLASH Option Bytes Programming functions: this group includes all needed functions to manage the Option Bytes:
  - Lock and Unlock the Option Bytes
  - Set/Reset the write protection
  - Set the Read protection Level
  - Program the user Option Bytes
  - Launch the Option Bytes loader
  - Erase Option Bytes
  - Program the data Option Bytes
  - Get the Write protection.
  - Get the user option bytes.
3. Interrupts and flags management functions : this group includes all needed functions to:
  - Handle FLASH interrupts
  - Wait for last FLASH operation according to its status
  - Get error flag status

In addition to these function, this driver includes a set of macros allowing to handle the following operations:

- Set/Get the latency
- Enable/Disable the prefetch buffer
- Enable/Disable the half cycle access
- Enable/Disable the FLASH interrupts
- Monitor the FLASH flags status

### 19.2.3 Peripheral Control functions

This subsection provides a set of functions allowing to control the FLASH memory operations.

This section contains the following APIs:

- [\*HAL\\_FLASH\\_Unlock\(\)\*](#)
- [\*HAL\\_FLASH\\_Lock\(\)\*](#)
- [\*HAL\\_FLASH\\_OB\\_Unlock\(\)\*](#)
- [\*HAL\\_FLASH\\_OB\\_Lock\(\)\*](#)
- [\*HAL\\_FLASH\\_OB\\_Launch\(\)\*](#)

### 19.2.4 Peripheral Errors functions

This subsection permit to get in run-time errors of the FLASH peripheral.

This section contains the following APIs:

- [\*HAL\\_FLASH\\_GetError\(\)\*](#)

## 19.2.5 Detailed description of functions

### HAL\_FLASH\_Program

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_FLASH_Program (uint32_t TypeProgram, uint32_t Address, uint64_t Data)</b>                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Program halfword, word or double word at a specified address.                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TypeProgram:</b> Indicate the way to program at a specified address. This parameter can be a value of FLASH Type Program</li> <li>• <b>Address:</b> Specifie the address to be programmed.</li> <li>• <b>Data:</b> Specifie the data to be programmed</li> </ul>                                                                                                                                                                                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL_StatusTypeDef:</b> HAL Status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• The function HAL_FLASH_Unlock() should be called before to unlock the FLASH interface The function HAL_FLASH_Lock() should be called after to lock the FLASH interface</li> <li>• If an erase and a program operations are requested simultaneously, the erase operation is performed before the program one.</li> <li>• FLASH should be previously erased before new programmation (only exception to this is when 0x0000 is programmed)</li> </ul> |

### HAL\_FLASH\_Program\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_FLASH_Program_IT (uint32_t TypeProgram, uint32_t Address, uint64_t Data)</b>                                                                                                                                                                                                                                                             |
| Function description | Program halfword, word or double word at a specified address with interrupt enabled.                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TypeProgram:</b> Indicate the way to program at a specified address. This parameter can be a value of FLASH Type Program</li> <li>• <b>Address:</b> Specifie the address to be programmed.</li> <li>• <b>Data:</b> Specifie the data to be programmed</li> </ul>                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL_StatusTypeDef:</b> HAL Status</li> </ul>                                                                                                                                                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• The function HAL_FLASH_Unlock() should be called before to unlock the FLASH interface The function HAL_FLASH_Lock() should be called after to lock the FLASH interface</li> <li>• If an erase and a program operations are requested simultaneously, the erase operation is performed before the program one.</li> </ul> |

### HAL\_FLASH\_IRQHandler

|                      |                                                                 |
|----------------------|-----------------------------------------------------------------|
| Function name        | <b>void HAL_FLASH_IRQHandler (void )</b>                        |
| Function description | This function handles FLASH interrupt request.                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |

**HAL\_FLASH\_EndOfOperationCallback**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_FLASH_EndOfOperationCallback (uint32_t ReturnValue)</b>                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | FLASH end of operation interrupt callback.                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ReturnValue:</b> The value saved in this parameter depends on the ongoing procedure           <ul style="list-style-type: none"> <li>– Mass Erase: No return value expected</li> <li>– Pages Erase: Address of the page which has been erased (if 0xFFFFFFFF, it means that all the selected pages have been erased)</li> <li>– Program: Address which was selected for data program</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                         |

**HAL\_FLASH\_OperationErrorHandler**

|                      |                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_FLASH_OperationErrorHandler (uint32_t ReturnValue)</b>                                                                                                                                                                                                                                                                                                                 |
| Function description | FLASH operation error interrupt callback.                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ReturnValue:</b> The value saved in this parameter depends on the ongoing procedure           <ul style="list-style-type: none"> <li>– Mass Erase: No return value expected</li> <li>– Pages Erase: Address of the page which returned an error</li> <li>– Program: Address which was selected for data program</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                    |

**HAL\_FLASH\_Unlock**

|                      |                                                                        |
|----------------------|------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_FLASH_Unlock (void )</b>                      |
| Function description | Unlock the FLASH control register access.                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> Status</li> </ul> |

**HAL\_FLASH\_Lock**

|                      |                                                                        |
|----------------------|------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_FLASH_Lock (void )</b>                        |
| Function description | Locks the FLASH control register access.                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> Status</li> </ul> |

**HAL\_FLASH\_OB\_Unlock**

|                      |                                                                        |
|----------------------|------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_FLASH_OB_Unlock (void )</b>                   |
| Function description | Unlock the FLASH Option Control Registers access.                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> Status</li> </ul> |

**HAL\_FLASH\_OB\_Lock**

Function name      **HAL\_StatusTypeDef HAL\_FLASH\_OB\_Lock (void )**

Function description      Lock the FLASH Option Control Registers access.

Return values      •    **HAL:** Status

**HAL\_FLASH\_OB\_Launch**

Function name      **HAL\_StatusTypeDef HAL\_FLASH\_OB\_Launch (void )**

Function description      Launch the option byte loading.

Return values      •    **HAL:** Status

Notes      •    This function will reset automatically the MCU.

**HAL\_FLASH\_GetError**

Function name      **uint32\_t HAL\_FLASH\_GetError (void )**

Function description      Get the specific FLASH error flag.

Return values      •    **FLASH\_ErrorCode:** The returned value can be: FLASH Error Codes

**FLASH\_WaitForLastOperation**

Function name      **HAL\_StatusTypeDef FLASH\_WaitForLastOperation (uint32\_t Timeout)**

Function description      Wait for a FLASH operation to complete.

Parameters      •    **Timeout:** maximum flash operation timeout

Return values      •    **HAL:** Status

## 19.3 FLASH Firmware driver defines

### 19.3.1 FLASH

***FLASH Error Codes***

**HAL\_FLASH\_ERROR\_NONE**      No error

**HAL\_FLASH\_ERROR\_PROG**      Programming error

**HAL\_FLASH\_ERROR\_WRP**      Write protection error

***FLASH Flag definition***

**FLASH\_FLAG\_BSY**      FLASH Busy flag

**FLASH\_FLAG\_PGERR**      FLASH Programming error flag

**FLASH\_FLAG\_WRPERR**      FLASH Write protected error flag

**FLASH\_FLAG\_EOP**      FLASH End of Operation flag

***FLASH Half Cycle***

`__HAL_FLASH_HALF_CYCLE_ACCESS_ENABLE`   **Description:**  
• Enable the FLASH half cycle access.

**Return value:**

- None

`__HAL_FLASH_HALF_CYCLE_ACCESS_DISABLE`   **Description:**  
• Disable the FLASH half cycle access.

**Return value:**

- None

***FLASH Interrupts***

`__HAL_FLASH_ENABLE_IT`   **Description:**  
• Enable the specified FLASH interrupt.

**Parameters:**

- `__INTERRUPT__`: FLASH interrupt This parameter can be any combination of the following values:
  - `FLASH_IT_EOP` End of FLASH Operation Interrupt
  - `FLASH_IT_ERR` Error Interrupt

**Return value:**

- none

`__HAL_FLASH_DISABLE_IT`   **Description:**  
• Disable the specified FLASH interrupt.

**Parameters:**

- `__INTERRUPT__`: FLASH interrupt This parameter can be any combination of the following values:
  - `FLASH_IT_EOP` End of FLASH Operation Interrupt
  - `FLASH_IT_ERR` Error Interrupt

**Return value:**

- none

`__HAL_FLASH_GET_FLAG`   **Description:**  
• Get the specified FLASH flag status.

**Parameters:**

- `__FLAG__`: specifies the FLASH flag to check. This parameter can be one of the following values:
  - `FLASH_FLAG_BSY` FLASH Busy flag
  - `FLASH_FLAG_EOP` FLASH End of Operation flag
  - `FLASH_FLAG_WRPERR` FLASH Write

- protected error flag
- FLASH\_FLAG\_PGERR FLASH Programming error flag

**Return value:**

- The new state of \_\_FLAG\_\_ (SET or RESET).

**\_HAL\_FLASH\_CLEAR\_FLAG****Description:**

- Clear the specified FLASH flag.

**Parameters:**

- \_\_FLAG\_\_: specifies the FLASH flags to clear. This parameter can be any combination of the following values:
  - FLASH\_FLAG\_EOP FLASH End of Operation flag
  - FLASH\_FLAG\_WRPERR FLASH Write protected error flag
  - FLASH\_FLAG\_PGERR FLASH Programming error flag

**Return value:**

- none

***FLASH Interrupt definition***

**FLASH\_IT\_EOP** End of FLASH Operation Interrupt source

**FLASH\_IT\_ERR** Error Interrupt source

***FLASH Latency***

**FLASH\_LATENCY\_0** FLASH Zero Latency cycle

**FLASH\_LATENCY\_1** FLASH One Latency cycle

**FLASH\_LATENCY\_2** FLASH Two Latency cycles

***FLASH Prefetch*****\_HAL\_FLASH\_PREFETCH\_BUFFER\_ENABLE** **Description:**

- Enable the FLASH prefetch buffer.

**Return value:**

- None

**\_HAL\_FLASH\_PREFETCH\_BUFFER\_DISABLE** **Description:**

- Disable the FLASH prefetch buffer.

**Return value:**

- None

***FLASH Type Program***

|                              |                                                       |
|------------------------------|-------------------------------------------------------|
| FLASH_TYPEPROGRAM_HALFWORD   | Program a half-word (16-bit) at a specified address.  |
| FLASH_TYPEPROGRAM_WORD       | Program a word (32-bit) at a specified address.       |
| FLASH_TYPEPROGRAM_DOUBLEWORD | Program a double word (64-bit) at a specified address |

## 20 HAL FLASH Extension Driver

### 20.1 FLASHEx Firmware driver registers structures

#### 20.1.1 FLASH\_EraseInitTypeDef

##### Data Fields

- *uint32\_t TypeErase*
- *uint32\_t PageAddress*
- *uint32\_t NbPages*

##### Field Documentation

- ***uint32\_t FLASH\_EraseInitTypeDef::TypeErase***  
TypeErase: Mass erase or page erase. This parameter can be a value of [\*\*FLASHEx\\_Type\\_Erase\*\*](#)
- ***uint32\_t FLASH\_EraseInitTypeDef::PageAddress***  
PageAddress: Initial FLASH page address to erase when mass erase is disabled. This parameter must be a number between Min\_Data = FLASH\_BASE and Max\_Data = FLASH\_BANK1\_END
- ***uint32\_t FLASH\_EraseInitTypeDef::NbPages***  
NbPages: Number of pages to be erased. This parameter must be a value between Min\_Data = 1 and Max\_Data = (max number of pages - value of initial page)

#### 20.1.2 FLASH\_OBProgramInitTypeDef

##### Data Fields

- *uint32\_t OptionType*
- *uint32\_t WRPState*
- *uint32\_t WRPPage*
- *uint8\_t RDPLevel*
- *uint8\_t USERConfig*
- *uint32\_t DATAAddress*
- *uint8\_t DATAData*

##### Field Documentation

- ***uint32\_t FLASH\_OBProgramInitTypeDef::OptionType***  
OptionType: Option byte to be configured. This parameter can be a value of [\*\*FLASHEx\\_OB\\_Type\*\*](#)
- ***uint32\_t FLASH\_OBProgramInitTypeDef::WRPState***  
WRPState: Write protection activation or deactivation. This parameter can be a value of [\*\*FLASHEx\\_OB\\_WRP\\_State\*\*](#)
- ***uint32\_t FLASH\_OBProgramInitTypeDef::WRPPage***  
WRPPage: specifies the page(s) to be write protected. This parameter can be a value of [\*\*FLASHEx\\_OB\\_Write\\_Protection\*\*](#)
- ***uint8\_t FLASH\_OBProgramInitTypeDef::RDPLevel***  
RDPLevel: Set the read protection level.. This parameter can be a value of [\*\*FLASHEx\\_OB\\_Read\\_Protection\*\*](#)
- ***uint8\_t FLASH\_OBProgramInitTypeDef::USERConfig***  
USERConfig: Program the FLASH User Option Byte: IWDG / STOP / STDBY / BOOT1 / VDDA\_ANALOG / SRAM\_PARITY / SDADC12\_VDD\_MONITOR. This parameter can be a combination of [\*\*FLASHEx\\_OB\\_IWatchdog\*\*](#).

- FLASHEx\_OB\_nRST\_STOP, FLASHEx\_OB\_nRST\_STDBY,  
 FLASHEx\_OB\_BOOT1, FLASHEx\_OB\_VDDA\_Analog\_Monitoring,  
 FLASHEx\_OB\_RAM\_Parity\_Check\_Enable.*
- ***uint32\_t FLASH\_OBProgramInitTypeDef::DATAAddress***  
 DATAAddress: Address of the option byte DATA to be programmed This parameter can be a value of *FLASHEx\_OB\_Data\_Address*
  - ***uint8\_t FLASH\_OBProgramInitTypeDef::DATAData***  
 DATAData: Data to be stored in the option byte DATA This parameter must be a number between Min\_Data = 0x00 and Max\_Data = 0xFFU

## 20.2 FLASHEx Firmware driver API description

### 20.2.1 FLASH Erasing Programming functions

The FLASH Memory Erasing functions, includes the following functions:

- *@ref HAL\_FLASHEx\_Erase*: return only when erase has been done
- *@ref HAL\_FLASHEx\_Erase\_IT*: end of erase is done when *@ref HAL\_FLASH\_EndOfOperationCallback* is called with parameter 0xFFFFFFFF

Any operation of erase should follow these steps:

1. Call the *@ref HAL\_FLASH\_Unlock()* function to enable the flash control register and program memory access.
2. Call the desired function to erase page.
3. Call the *@ref HAL\_FLASH\_Lock()* to disable the flash program memory access (recommended to protect the FLASH memory against possible unwanted operation).

This section contains the following APIs:

- ***HAL\_FLASHEx\_Erase()***
- ***HAL\_FLASHEx\_Erase\_IT()***

### 20.2.2 Option Bytes Programming functions

This subsection provides a set of functions allowing to control the FLASH option bytes operations.

This section contains the following APIs:

- ***HAL\_FLASHEx\_OBErase()***
- ***HAL\_FLASHEx\_OBProgram()***
- ***HAL\_FLASHEx\_OBGetConfig()***
- ***HAL\_FLASHEx\_OBGetUserData()***

### 20.2.3 Detailed description of functions

#### HAL\_FLASHEx\_Erase

Function name      **HAL\_StatusTypeDef HAL\_FLASHEx\_Erase  
 (FLASH\_EraselInitTypeDef \* pEraselInit, uint32\_t \* PageError)**

Function description      Perform a mass erase or erase the specified FLASH memory pages.

Parameters     
 

- **pEraselInit**: pointer to an *FLASH\_EraselInitTypeDef* structure that contains the configuration information for the erasing.
- **PageError**: pointer to variable that contains the configuration information on faulty page in case of error (0xFFFFFFFF)

---

|               |                                                                                                                                                                                                                                                                                       |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | means that all the pages have been correctly erased)                                                                                                                                                                                                                                  |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL_StatusTypeDef:</b> HAL Status</li> </ul>                                                                                                                                                                                              |
| Notes         | <ul style="list-style-type: none"> <li>• To correctly run this function, the HAL_FLASH_Unlock() function must be called before. Call the HAL_FLASH_Lock() to disable the flash memory access (recommended to protect the FLASH memory against possible unwanted operation)</li> </ul> |

### **HAL\_FLASHEx\_Erase\_IT**

|                      |                                                                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_FLASHEx_Erase_IT<br/>(FLASH_EraseInitTypeDef * pEraseInit)</b>                                                                                                                                                                                               |
| Function description | Perform a mass erase or erase the specified FLASH memory pages with interrupt enabled.                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>pEraseInit:</b> pointer to an FLASH_EraseInitTypeDef structure that contains the configuration information for the erasing.</li> </ul>                                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL_StatusTypeDef:</b> HAL Status</li> </ul>                                                                                                                                                                                              |
| Notes                | <ul style="list-style-type: none"> <li>• To correctly run this function, the HAL_FLASH_Unlock() function must be called before. Call the HAL_FLASH_Lock() to disable the flash memory access (recommended to protect the FLASH memory against possible unwanted operation)</li> </ul> |

### **HAL\_FLASHEx\_OBErase**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_FLASHEx_OBErase (void )</b>                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Erases the FLASH option bytes.                                                                                                                                                                                                                                                                                                                                                                                                         |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                 |
| Notes                | <ul style="list-style-type: none"> <li>• This functions erases all option bytes except the Read protection (RDP). The function HAL_FLASH_Unlock() should be called before to unlock the FLASH interface The function HAL_FLASH_OB_Unlock() should be called before to unlock the options bytes The function HAL_FLASH_OB_Launch() should be called after to force the reload of the options bytes (system reset will occur)</li> </ul> |

### **HAL\_FLASHEx\_OBProgram**

|                      |                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_FLASHEx_OBProgram<br/>(FLASH_OBProgramInitTypeDef * pOBInit)</b>                                                                                                                                                                                                                                           |
| Function description | Program option bytes.                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>pOBInit:</b> pointer to an FLASH_OBInitStruct structure that contains the configuration information for the programming.</li> </ul>                                                                                                                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL_StatusTypeDef:</b> HAL Status</li> </ul>                                                                                                                                                                                                                                            |
| Notes                | <ul style="list-style-type: none"> <li>• The function HAL_FLASH_Unlock() should be called before to unlock the FLASH interface The function HAL_FLASH_OB_Unlock() should be called before to unlock the options bytes The function HAL_FLASH_OB_Launch() should be called after to force the reload of the options bytes</li> </ul> |

(system reset will occur)

**HAL\_FLASHEx\_OBGetConfig**

|                      |                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_FLASHEx_OBGetConfig<br/>(FLASH_OBProgramInitTypeDef * pOBInit)</b>                                                                                                  |
| Function description | Get the Option byte configuration.                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>pOBInit:</b> pointer to an FLASH_OBInitStruct structure that contains the configuration information for the programming.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                 |

**HAL\_FLASHEx\_OBGetUserData**

|                      |                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_FLASHEx_OBGetUserData (uint32_t DATAAddress)</b>                                                                                                                                                                                                       |
| Function description | Get the Option byte user data.                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DATAAddress:</b> Address of the option byte DATA This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– OB_DATA_ADDRESS_DATA0</li> <li>– OB_DATA_ADDRESS_DATA1</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Value:</b> programmed in USER data</li> </ul>                                                                                                                                                                              |

**20.3 FLASHEx Firmware driver defines****20.3.1 FLASHEx*****Option Byte BOOT1***

|                       |             |
|-----------------------|-------------|
| <b>OB_BOOT1_RESET</b> | BOOT1 Reset |
| <b>OB_BOOT1_SET</b>   | BOOT1 Set   |

***Option Byte Data Address***

|                              |  |
|------------------------------|--|
| <b>OB_DATA_ADDRESS_DATA0</b> |  |
| <b>OB_DATA_ADDRESS_DATA1</b> |  |

***Option Byte IWatchdog***

|                   |                        |
|-------------------|------------------------|
| <b>OB_IWDG_SW</b> | Software IWDG selected |
| <b>OB_IWDG_HW</b> | Hardware IWDG selected |

***Option Byte nRST STDBY***

|                        |                                             |
|------------------------|---------------------------------------------|
| <b>OB_STDBY_NO_RST</b> | No reset generated when entering in STANDBY |
| <b>OB_STDBY_RST</b>    | Reset generated when entering in STANDBY    |

***Option Byte nRST STOP***

|                       |                                          |
|-----------------------|------------------------------------------|
| <b>OB_STOP_NO_RST</b> | No reset generated when entering in STOP |
| <b>OB_STOP_RST</b>    | Reset generated when entering in STOP    |

***Option Byte SRAM Parity Check Enable***

OB\_SRAM\_PARITY\_SET      SRAM parity check enable set

OB\_SRAM\_PARITY\_RESET    SRAM parity check enable reset

***Option Byte Read Protection***

OB\_RDP\_LEVEL\_0

OB\_RDP\_LEVEL\_1

OB\_RDP\_LEVEL\_2      Warning: When enabling read protection level 2 it's no more possible to go back to level 1 or 0U

***Option Bytes Type***

OPTIONBYTE\_WRP      WRP option byte configuration

OPTIONBYTE\_RDP      RDP option byte configuration

OPTIONBYTE\_USER      USER option byte configuration

OPTIONBYTE\_DATA      DATA option byte configuration

***Option Byte VDDA Analog Monitoring***

OB\_VDDA\_ANALOG\_ON      Analog monitoring on VDDA Power source ON

OB\_VDDA\_ANALOG\_OFF      Analog monitoring on VDDA Power source OFF

***FLASHEx OB Write Protection***

OB\_WRP\_PAGES0TO1

OB\_WRP\_PAGES2TO3

OB\_WRP\_PAGES4TO5

OB\_WRP\_PAGES6TO7

OB\_WRP\_PAGES8TO9

OB\_WRP\_PAGES10TO11

OB\_WRP\_PAGES12TO13

OB\_WRP\_PAGES14TO15

OB\_WRP\_PAGES16TO17

OB\_WRP\_PAGES18TO19

OB\_WRP\_PAGES20TO21

OB\_WRP\_PAGES22TO23

OB\_WRP\_PAGES24TO25

OB\_WRP\_PAGES26TO27

OB\_WRP\_PAGES28TO29

OB\_WRP\_PAGES30TO31

OB\_WRP\_PAGES32TO33

OB\_WRP\_PAGES34TO35

OB\_WRP\_PAGES36TO37

OB\_WRP\_PAGES38TO39

OB\_WRP\_PAGES40TO41  
OB\_WRP\_PAGES42TO43  
OB\_WRP\_PAGES44TO45  
OB\_WRP\_PAGES46TO47  
OB\_WRP\_PAGES48TO49  
OB\_WRP\_PAGES50TO51  
OB\_WRP\_PAGES52TO53  
OB\_WRP\_PAGES54TO55  
OB\_WRP\_PAGES56TO57  
OB\_WRP\_PAGES58TO59  
OB\_WRP\_PAGES60TO61  
OB\_WRP\_PAGES62TO127  
OB\_WRP\_PAGES0TO15MASK  
OB\_WRP\_PAGES16TO31MASK  
OB\_WRP\_PAGES32TO47MASK  
OB\_WRP\_PAGES32TO47MASK  
OB\_WRP\_PAGES48TO127MASK  
OB\_WRP\_PAGES48TO127MASK

**OB\_WRP\_ALLPAGES** Write protection of all pages

## ***Option Byte WRP State***

**OB\_WRPSTATE\_DISABLE** Disable the write protection of the desired pages

**OB\_WRPSTATE\_ENABLE** Enable the write protection of the desired pages

## *FLASHEx Page Size*

## FLASH\_PAGE\_SIZE

## *FLASH Type Erase*

**FLASH\_TYPEERASE\_PAGES** Pages erase only

**FLASH\_TYPEERASE\_MASSERASE** Flash mass erase activation

## 21 HAL GPIO Generic Driver

### 21.1 GPIO Firmware driver registers structures

#### 21.1.1 GPIO\_InitTypeDef

##### Data Fields

- *uint32\_t Pin*
- *uint32\_t Mode*
- *uint32\_t Pull*
- *uint32\_t Speed*
- *uint32\_t Alternate*

##### Field Documentation

- ***uint32\_t GPIO\_InitTypeDef::Pin***  
Specifies the GPIO pins to be configured. This parameter can be any value of [\*\*GPIO\\_pins\*\*](#)
- ***uint32\_t GPIO\_InitTypeDef::Mode***  
Specifies the operating mode for the selected pins. This parameter can be a value of [\*\*GPIO\\_mode\*\*](#)
- ***uint32\_t GPIO\_InitTypeDef::Pull***  
Specifies the Pull-up or Pull-Down activation for the selected pins. This parameter can be a value of [\*\*GPIO\\_pull\*\*](#)
- ***uint32\_t GPIO\_InitTypeDef::Speed***  
Specifies the speed for the selected pins. This parameter can be a value of [\*\*GPIO\\_speed\*\*](#)
- ***uint32\_t GPIO\_InitTypeDef::Alternate***  
Peripheral to be connected to the selected pins This parameter can be a value of [\*\*GPIOEx\\_Alternate\\_function\\_selection\*\*](#)

### 21.2 GPIO Firmware driver API description

#### 21.2.1 GPIO Peripheral features

- Each port bit of the general-purpose I/O (GPIO) ports can be individually configured by software in several modes:
  - Input mode
  - Analog mode
  - Output mode
  - Alternate function mode
  - External interrupt/event lines
- During and just after reset, the alternate functions and external interrupt lines are not active and the I/O ports are configured in input floating mode.
- All GPIO pins have weak internal pull-up and pull-down resistors, which can be activated or not.
- In Output or Alternate mode, each IO can be configured on open-drain or push-pull type and the IO speed can be selected depending on the VDD value.
- The microcontroller IO pins are connected to onboard peripherals/modules through a multiplexer that allows only one peripheral alternate function (AF) connected to an IO pin at a time. In this way, there can be no conflict between peripherals sharing the same IO pin.

- All ports have external interrupt/event capability. To use external interrupt lines, the port must be configured in input mode. All available GPIO pins are connected to the 16 external interrupt/event lines from EXTI0 to EXTI15.
- The external interrupt/event controller consists of up to 23 edge detectors (16 lines are connected to GPIO) for generating event/interrupt requests (each input line can be independently configured to select the type (interrupt or event) and the corresponding trigger event (rising or falling or both). Each line can also be masked independently).

### 21.2.2 How to use this driver

1. Enable the GPIO AHB clock using the following function:  
`__HAL_RCC_GPIOx_CLK_ENABLE()`.
2. Configure the GPIO pin(s) using `HAL_GPIO_Init()`.
  - Configure the IO mode using "Mode" member from `GPIO_InitTypeDef` structure
  - Activate Pull-up, Pull-down resistor using "Pull" member from `GPIO_InitTypeDef` structure.
  - In case of Output or alternate function mode selection: the speed is configured through "Speed" member from `GPIO_InitTypeDef` structure.
  - In alternate mode is selection, the alternate function connected to the IO is configured through "Alternate" member from `GPIO_InitTypeDef` structure.
  - Analog mode is required when a pin is to be used as ADC channel or DAC output.
  - In case of external interrupt/event selection the "Mode" member from `GPIO_InitTypeDef` structure select the type (interrupt or event) and the corresponding trigger event (rising or falling or both).
3. In case of external interrupt/event mode selection, configure NVIC IRQ priority mapped to the EXTI line using `HAL_NVIC_SetPriority()` and enable it using `HAL_NVIC_EnableIRQ()`.
4. To get the level of a pin configured in input mode use `HAL_GPIO_ReadPin()`.
5. To set/reset the level of a pin configured in output mode use `HAL_GPIO_WritePin()`/`HAL_GPIO_TogglePin()`.
6. To lock pin configuration until next reset use `HAL_GPIO_LockPin()`.
7. During and just after reset, the alternate functions are not active and the GPIO pins are configured in input floating mode (except JTAG pins).
8. The LSE oscillator pins OSC32\_IN and OSC32\_OUT can be used as general purpose (PC14 and PC15U, respectively) when the LSE oscillator is off. The LSE has priority over the GPIO function.
9. The HSE oscillator pins OSC\_IN/OSC\_OUT can be used as general purpose PF0 and PF1, respectively, when the HSE oscillator is off. The HSE has priority over the GPIO function.

### 21.2.3 Initialization and de-initialization functions

This section contains the following APIs:

- `HAL_GPIO_Init()`
- `HAL_GPIO_DeInit()`

### 21.2.4 IO operation functions

This section contains the following APIs:

- `HAL_GPIO_ReadPin()`
- `HAL_GPIO_WritePin()`
- `HAL_GPIO_TogglePin()`
- `HAL_GPIO_LockPin()`
- `HAL_GPIO_EXTI_IRQHandler()`

- [\*HAL\\_GPIO\\_EXTI\\_Callback\(\)\*](#)

## 21.2.5 Detailed description of functions

### **HAL\_GPIO\_Init**

|                      |                                                                                                                                                                                                                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_GPIO_Init (GPIO_TypeDef * GPIOx,<br/>GPIO_InitTypeDef * GPIO_InitStruct)</b>                                                                                                                                                                                                                   |
| Function description | Initialize the GPIOx peripheral according to the specified parameters in the GPIO_Init.                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> where x can be (A..F) to select the GPIO peripheral for STM32F3 family devices</li> <li>• <b>GPIO_InitStruct:</b> pointer to a GPIO_InitTypeDef structure that contains the configuration information for the specified GPIO peripheral.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                            |

### **HAL\_GPIO\_DeInit**

|                      |                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_GPIO_DeInit (GPIO_TypeDef * GPIOx, uint32_t GPIO_Pin)</b>                                                                                                                                                                                                                            |
| Function description | De-initialize the GPIOx peripheral registers to their default reset values.                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> where x can be (A..F) to select the GPIO peripheral for STM32F30X device or STM32F37X device</li> <li>• <b>GPIO_Pin:</b> specifies the port bit to be written. This parameter can be one of GPIO_PIN_x where x can be (0..15).</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                  |

### **HAL\_GPIO\_ReadPin**

|                      |                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>GPIO_PinState HAL_GPIO_ReadPin (GPIO_TypeDef * GPIOx,<br/>uint16_t GPIO_Pin)</b>                                                                                                                                                                           |
| Function description | Read the specified input port pin.                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> where x can be (A..F) to select the GPIO peripheral for STM32F3 family</li> <li>• <b>GPIO_Pin:</b> specifies the port bit to read. This parameter can be GPIO_PIN_x where x can be (0..15).</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>The:</b> input port pin value.</li> </ul>                                                                                                                                                                         |

### **HAL\_GPIO\_WritePin**

|                      |                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_GPIO_WritePin (GPIO_TypeDef * GPIOx, uint16_t GPIO_Pin, GPIO_PinState PinState)</b>                                                                                                                                                                            |
| Function description | Set or clear the selected data port bit.                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> where x can be (A..F) to select the GPIO peripheral for STM32F3 family</li> <li>• <b>GPIO_Pin:</b> specifies the port bit to be written. This parameter can be one of GPIO_PIN_x where x can be (0..15).</li> </ul> |

|               |                                                                                                                                                                                                                                                                                                                                           |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>• <b>PinState:</b> specifies the value to be written to the selected bit. This parameter can be one of the GPIO_PinState enum values:           <ul style="list-style-type: none"> <li>– GPIO_PIN_RESET: to clear the port pin</li> <li>– GPIO_PIN_SET: to set the port pin</li> </ul> </li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                           |
| Notes         | <ul style="list-style-type: none"> <li>• This function uses GPIOx_BSRR and GPIOx_BRR registers to allow atomic read/modify accesses. In this way, there is no risk of an IRQ occurring between the read and the modify access.</li> </ul>                                                                                                 |

### HAL\_GPIO\_TogglePin

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_GPIO_TogglePin (GPIO_TypeDef * GPIOx, uint16_t GPIO_Pin)</b>                                                                                                                              |
| Function description | Toggle the specified GPIO pin.                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> where x can be (A..F) to select the GPIO peripheral for STM32F3 family</li> <li>• <b>GPIO_Pin:</b> specifies the pin to be toggled.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |

### HAL\_GPIO\_LockPin

|                      |                                                                                                                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_GPIO_LockPin (GPIO_TypeDef * GPIOx, uint16_t GPIO_Pin)</b>                                                                                                                                                                                                    |
| Function description | Lock GPIO Pins configuration registers.                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> where x can be (A..F) to select the GPIO peripheral for STM32F3 family</li> <li>• <b>GPIO_Pin:</b> specifies the port bits to be locked. This parameter can be any combination of GPIO_Pin_x where x can be (0..15).</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• The locked registers are GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR, GPIOx_AFRL and GPIOx_AFRH.</li> <li>• The configuration of the locked GPIO pins can no longer be modified until the next reset.</li> </ul>                    |

### HAL\_GPIO\_EXTI\_IRQHandler

|                      |                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_GPIO_EXTI_IRQHandler (uint16_t GPIO_Pin)</b>                                                                          |
| Function description | Handle EXTI interrupt request.                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIO_Pin:</b> Specifies the port pin connected to corresponding EXTI line.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                   |

**HAL\_GPIO\_EXTI\_Callback**

|                      |                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_GPIO_EXTI_Callback (uint16_t GPIO_Pin)</b>                                                                            |
| Function description | EXTI line detection callback.                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIO_Pin:</b> Specifies the port pin connected to corresponding EXTI line.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                   |

**21.3 GPIO Firmware driver defines****21.3.1 GPIO*****GPIO Exported Macros*****\_HAL\_GPIO\_EXTI\_GET\_FLAG****Description:**

- Check whether the specified EXTI line flag is set or not.

**Parameters:**

- \_EXTI\_LINE\_: specifies the EXTI line flag to check. This parameter can be GPIO\_PIN\_x where x can be(0..15)

**Return value:**

- The: new state of \_EXTI\_LINE\_ (SET or RESET).

**\_HAL\_GPIO\_EXTI\_CLEAR\_FLAG****Description:**

- Clear the EXTI's line pending flags.

**Parameters:**

- \_EXTI\_LINE\_: specifies the EXTI lines flags to clear. This parameter can be any combination of GPIO\_PIN\_x where x can be (0..15)

**Return value:**

- None

**\_HAL\_GPIO\_EXTI\_GET\_IT****Description:**

- Check whether the specified EXTI line is asserted or not.

**Parameters:**

- \_EXTI\_LINE\_: specifies the EXTI line to check. This parameter can be GPIO\_PIN\_x where x can be(0..15)

**Return value:**

- The: new state of \_EXTI\_LINE\_ (SET or RESET).

[\\_\\_HAL\\_GPIO\\_EXTI\\_CLEAR\\_IT](#)**Description:**

- Clear the EXTI's line pending bits.

**Parameters:**

- \_\_EXTI\_LINE\_\_: specifies the EXTI lines to clear. This parameter can be any combination of GPIO\_PIN\_x where x can be (0..15)

**Return value:**

- None

[\\_\\_HAL\\_GPIO\\_EXTI\\_GENERATE\\_SWIT](#)**Description:**

- Generate a Software interrupt on selected EXTI line.

**Parameters:**

- \_\_EXTI\_LINE\_\_: specifies the EXTI line to check. This parameter can be GPIO\_PIN\_x where x can be(0..15)

**Return value:**

- None

***GPIO mode***

|                                              |                                                                    |
|----------------------------------------------|--------------------------------------------------------------------|
| <a href="#">GPIO_MODE_INPUT</a>              | Input Floating Mode                                                |
| <a href="#">GPIO_MODE_OUTPUT_PP</a>          | Output Push Pull Mode                                              |
| <a href="#">GPIO_MODE_OUTPUT_OD</a>          | Output Open Drain Mode                                             |
| <a href="#">GPIO_MODE_AF_PP</a>              | Alternate Function Push Pull Mode                                  |
| <a href="#">GPIO_MODE_AF_OD</a>              | Alternate Function Open Drain Mode                                 |
| <a href="#">GPIO_MODE_ANALOG</a>             | Analog Mode                                                        |
| <a href="#">GPIO_MODE_IT_RISING</a>          | External Interrupt Mode with Rising edge trigger detection         |
| <a href="#">GPIO_MODE_IT_FALLING</a>         | External Interrupt Mode with Falling edge trigger detection        |
| <a href="#">GPIO_MODE_IT_RISING_FALLING</a>  | External Interrupt Mode with Rising/Falling edge trigger detection |
| <a href="#">GPIO_MODE_EVT_RISING</a>         | External Event Mode with Rising edge trigger detection             |
| <a href="#">GPIO_MODE_EVT_FALLING</a>        | External Event Mode with Falling edge trigger detection            |
| <a href="#">GPIO_MODE_EVT_RISING_FALLING</a> | External Event Mode with Rising/Falling edge trigger detection     |

***GPIO pins***

|                            |
|----------------------------|
| <a href="#">GPIO_PIN_0</a> |
| <a href="#">GPIO_PIN_1</a> |

GPIO\_PIN\_2  
GPIO\_PIN\_3  
GPIO\_PIN\_4  
GPIO\_PIN\_5  
GPIO\_PIN\_6  
GPIO\_PIN\_7  
GPIO\_PIN\_8  
GPIO\_PIN\_9  
GPIO\_PIN\_10  
GPIO\_PIN\_11  
GPIO\_PIN\_12  
GPIO\_PIN\_13  
GPIO\_PIN\_14  
GPIO\_PIN\_15  
GPIO\_PIN\_All  
GPIO\_PIN\_MASK

***GPIO pull***

GPIO\_NOPULL      No Pull-up or Pull-down activation  
GPIO\_PULLUP      Pull-up activation  
GPIO\_PULLDOWN    Pull-down activation

***GPIO speed***

|                        |                                                               |
|------------------------|---------------------------------------------------------------|
| GPIO_SPEED_FREQ_LOW    | range up to 2 MHz, please refer to the product datasheet      |
| GPIO_SPEED_FREQ_MEDIUM | range 4 MHz to 10 MHz, please refer to the product datasheet  |
| GPIO_SPEED_FREQ_HIGH   | range 10 MHz to 50 MHz, please refer to the product datasheet |

## 22 HAL GPIO Extension Driver

### 22.1 GPIOEx Firmware driver defines

#### 22.1.1 GPIOEx

##### *GPIOEx Alternate function selection*

GPIO\_AF0\_RTC\_50Hz  
GPIO\_AF0\_MCO  
GPIO\_AF0\_TAMPER  
GPIO\_AF0\_SWJ  
GPIO\_AF0\_TRACE  
GPIO\_AF1\_TIM2  
GPIO\_AF1\_TIM5  
GPIO\_AF1\_TIM16  
GPIO\_AF1\_TIM17  
GPIO\_AF1\_EVENTOUT  
GPIO\_AF2\_TIM1  
GPIO\_AF2\_TIM2  
GPIO\_AF2\_TIM3  
GPIO\_AF2\_TIM4  
GPIO\_AF2\_TIM8  
GPIO\_AF2\_TIM15  
GPIO\_AF2\_COMP1  
GPIO\_AF3\_TSC  
GPIO\_AF3\_TIM8  
GPIO\_AF3\_COMP7  
GPIO\_AF3\_TIM15  
GPIO\_AF4\_TIM1  
GPIO\_AF4\_TIM8  
GPIO\_AF4\_TIM16  
GPIO\_AF4\_TIM17  
GPIO\_AF4\_I2C1  
GPIO\_AF4\_I2C2  
GPIO\_AF5\_SPI1  
GPIO\_AF5\_SPI2  
GPIO\_AF5\_SPI3

GPIO\_AF5\_I2S  
GPIO\_AF5\_I2S2ext  
GPIO\_AF5\_TIM8  
GPIO\_AF5\_IR  
GPIO\_AF5\_UART4  
GPIO\_AF5\_UART5  
GPIO\_AF6\_SPI2  
GPIO\_AF6\_SPI3  
GPIO\_AF6\_I2S3ext  
GPIO\_AF6\_TIM1  
GPIO\_AF6\_TIM8  
GPIO\_AF6\_IR  
GPIO\_AF7\_USART1  
GPIO\_AF7\_USART2  
GPIO\_AF7\_USART3  
GPIO\_AF7\_COMP3  
GPIO\_AF7\_COMP5  
GPIO\_AF7\_COMP6  
GPIO\_AF7\_CAN  
GPIO\_AF8\_COMP1  
GPIO\_AF8\_COMP2  
GPIO\_AF8\_COMP3  
GPIO\_AF8\_COMP4  
GPIO\_AF8\_COMP5  
GPIO\_AF8\_COMP6  
GPIO\_AF9\_CAN  
GPIO\_AF9\_TIM1  
GPIO\_AF9\_TIM8  
GPIO\_AF9\_TIM15  
GPIO\_AF10\_TIM2  
GPIO\_AF10\_TIM3  
GPIO\_AF10\_TIM4  
GPIO\_AF10\_TIM8  
GPIO\_AF10\_TIM17  
GPIO\_AF11\_TIM1  
GPIO\_AF11\_TIM8

GPIO\_AF12\_TIM1

GPIO\_AF14\_USB

GPIO\_AF15\_EVENTOUT

IS\_GPIO\_AF

***GPIOEx\_Get Port Index***

GPIO\_GET\_INDEX

## 23 HAL HRTIM Generic Driver

### 23.1 HRTIM Firmware driver registers structures

#### 23.1.1 HRTIM\_InitTypeDef

##### Data Fields

- *uint32\_t HRTIMInterruptRequests*
- *uint32\_t SyncOptions*
- *uint32\_t SyncInputSource*
- *uint32\_t SyncOutputSource*
- *uint32\_t SyncOutputPolarity*

##### Field Documentation

- *uint32\_t HRTIM\_InitTypeDef::HRTIMInterruptRequests*  
Specifies which interrupts requests must enabled for the HRTIM instance. This parameter can be any combination of [\*HRTIM\\_Common Interrupt Enable\*](#)
- *uint32\_t HRTIM\_InitTypeDef::SyncOptions*  
Specifies how the HRTIM instance handles the external synchronization signals. The HRTIM instance can be configured to act as a slave (waiting for a trigger to be synchronized) or a master (generating a synchronization signal) or both. This parameter can be a combination of [\*HRTIM\\_Synchronization Options\*](#).
- *uint32\_t HRTIM\_InitTypeDef::SyncInputSource*  
Specifies the external synchronization input source (significant only when the HRTIM instance is configured as a slave). This parameter can be a value of [\*HRTIM\\_Synchronization Input Source\*](#).
- *uint32\_t HRTIM\_InitTypeDef::SyncOutputSource*  
Specifies the source and event to be sent on the external synchronization outputs (significant only when the HRTIM instance is configured as a master). This parameter can be a value of [\*HRTIM\\_Synchronization Output Source\*](#)
- *uint32\_t HRTIM\_InitTypeDef::SyncOutputPolarity*  
Specifies the conditioning of the event to be sent on the external synchronization outputs (significant only when the HRTIM instance is configured as a master). This parameter can be a value of [\*HRTIM\\_Synchronization Output Polarity\*](#)

#### 23.1.2 HRTIM\_TimerParamTypeDef

##### Data Fields

- *uint32\_t CaptureTrigger1*
- *uint32\_t CaptureTrigger2*
- *uint32\_t InterruptRequests*
- *uint32\_t DMARequests*
- *uint32\_t DMASrcAddress*
- *uint32\_t DMADstAddress*
- *uint32\_t DMASize*

##### Field Documentation

- *uint32\_t HRTIM\_TimerParamTypeDef::CaptureTrigger1*  
Event(s) triggering capture unit 1. When the timer operates in Simple mode, this parameter can be a value of [\*HRTIM\\_External\\_Event\\_Channels\*](#). When the timer

- operates in Waveform mode, this parameter can be a combination of [\*\*HRTIM\\_Capture\\_Unit\\_Trigger\*\*](#).
- ***uint32\_t HRTIM\_TimerParamTypeDef::CaptureTrigger2***  
Event(s) triggering capture unit 2. When the timer operates in Simple mode, this parameter can be a value of [\*\*HRTIM\\_External\\_Event\\_Channels\*\*](#). When the timer operates in Waveform mode, this parameter can be a combination of [\*\*HRTIM\\_Capture\\_Unit\\_Trigger\*\*](#).
- ***uint32\_t HRTIM\_TimerParamTypeDef::InterruptRequests***  
Interrupts requests enabled for the timer.
- ***uint32\_t HRTIM\_TimerParamTypeDef::DMARequests***  
DMA requests enabled for the timer.
- ***uint32\_t HRTIM\_TimerParamTypeDef::DMASrcAddress***  
Address of the source address of the DMA transfer.
- ***uint32\_t HRTIM\_TimerParamTypeDef::DMADstAddress***  
Address of the destination address of the DMA transfer.
- ***uint32\_t HRTIM\_TimerParamTypeDef::DMASize***  
Size of the DMA transfer

### 23.1.3 [\\_\\_HRTIM\\_HandleTypeDef](#)

#### Data Fields

- ***HRTIM\_TypeDef \* Instance***
- ***HRTIM\_InitTypeDef Init***
- ***HRTIM\_TimerParamTypeDef TimerParam***
- ***HAL\_LockTypeDef Lock***
- ***\_IO HAL\_HRTIM\_StateTypeDef State***
- ***DMA\_HandleTypeDef \* hdmaMaster***
- ***DMA\_HandleTypeDef \* hdmaTimerA***
- ***DMA\_HandleTypeDef \* hdmaTimerB***
- ***DMA\_HandleTypeDef \* hdmaTimerC***
- ***DMA\_HandleTypeDef \* hdmaTimerD***
- ***DMA\_HandleTypeDef \* hdmaTimerE***

#### Field Documentation

- ***HRTIM\_TypeDef\* \_\_HRTIM\_HandleTypeDef::Instance***  
Register base address
- ***HRTIM\_InitTypeDef \_\_HRTIM\_HandleTypeDef::Init***  
HRTIM required parameters
- ***HRTIM\_TimerParamTypeDef \_\_HRTIM\_HandleTypeDef::TimerParam[MAX\_HRTIM\_TIMER]***  
HRTIM timers - including the master - parameters
- ***HAL\_LockTypeDef \_\_HRTIM\_HandleTypeDef::Lock***  
Locking object
- ***\_IO HAL\_HRTIM\_StateTypeDef \_\_HRTIM\_HandleTypeDef::State***  
HRTIM communication state
- ***DMA\_HandleTypeDef\* \_\_HRTIM\_HandleTypeDef::hdmaMaster***  
Master timer DMA handle parameters
- ***DMA\_HandleTypeDef\* \_\_HRTIM\_HandleTypeDef::hdmaTimerA***  
Timer A DMA handle parameters
- ***DMA\_HandleTypeDef\* \_\_HRTIM\_HandleTypeDef::hdmaTimerB***  
Timer B DMA handle parameters
- ***DMA\_HandleTypeDef\* \_\_HRTIM\_HandleTypeDef::hdmaTimerC***  
Timer C DMA handle parameters

- **DMA\_HandleTypeDef\* \_\_HRTIM\_HandleTypeDef::hdmaTimerD**  
Timer D DMA handle parameters
- **DMA\_HandleTypeDef\* \_\_HRTIM\_HandleTypeDef::hdmaTimerE**  
Timer E DMA handle parameters

### 23.1.4 HRTIM\_TimeBaseCfgTypeDef

#### Data Fields

- **uint32\_t Period**
- **uint32\_t RepetitionCounter**
- **uint32\_t PrescalerRatio**
- **uint32\_t Mode**

#### Field Documentation

- **uint32\_t HRTIM\_TimeBaseCfgTypeDef::Period**  
Specifies the timer period. The period value must be above 3 periods of the fHRTIM clock. Maximum value is = 0xFFFFFU
- **uint32\_t HRTIM\_TimeBaseCfgTypeDef::RepetitionCounter**  
Specifies the timer repetition period. This parameter must be a number between Min\_Data = 0x00 and Max\_Data = 0xFF.
- **uint32\_t HRTIM\_TimeBaseCfgTypeDef::PrescalerRatio**  
Specifies the timer clock prescaler ratio. This parameter can be any value of [HRTIM\\_Prescaler\\_Ratio](#)
- **uint32\_t HRTIM\_TimeBaseCfgTypeDef::Mode**  
Specifies the counter operating mode. This parameter can be any value of [HRTIM\\_Counter\\_Operating\\_Mode](#)

### 23.1.5 HRTIM\_SimpleOCChannelCfgTypeDef

#### Data Fields

- **uint32\_t Mode**
- **uint32\_t Pulse**
- **uint32\_t Polarity**
- **uint32\_t IdleLevel**

#### Field Documentation

- **uint32\_t HRTIM\_SimpleOCChannelCfgTypeDef::Mode**  
Specifies the output compare mode (toggle, active, inactive). This parameter can be any value of [HRTIM\\_Simple\\_OC\\_Mode](#)
- **uint32\_t HRTIM\_SimpleOCChannelCfgTypeDef::Pulse**  
Specifies the compare value to be loaded into the Compare Register. The compare value must be above or equal to 3 periods of the fHRTIM clock
- **uint32\_t HRTIM\_SimpleOCChannelCfgTypeDef::Polarity**  
Specifies the output polarity. This parameter can be any value of [HRTIM\\_Output\\_Polarity](#)
- **uint32\_t HRTIM\_SimpleOCChannelCfgTypeDef::IdleLevel**  
Specifies whether the output level is active or inactive when in IDLE state. This parameter can be any value of [HRTIM\\_Output\\_IDLE\\_Level](#)

### 23.1.6 HRTIM\_SimplePWMChannelCfgTypeDef

#### Data Fields

- **uint32\_t Pulse**
- **uint32\_t Polarity**

- *uint32\_t IdleLevel*

#### Field Documentation

- *uint32\_t HRTIM\_SimplePWMChannelCfgTypeDef::Pulse*  
Specifies the compare value to be loaded into the Compare Register. The compare value must be above or equal to 3 periods of the fHRTIM clock
- *uint32\_t HRTIM\_SimplePWMChannelCfgTypeDef::Polarity*  
Specifies the output polarity. This parameter can be any value of [HRTIM\\_Output\\_Polarity](#)
- *uint32\_t HRTIM\_SimplePWMChannelCfgTypeDef::IdleLevel*  
Specifies whether the output level is active or inactive when in IDLE state. This parameter can be any value of [HRTIM\\_Output\\_IDLE\\_Level](#)

### 23.1.7 HRTIM\_SimpleCaptureChannelCfgTypeDef

#### Data Fields

- *uint32\_t Event*
- *uint32\_t EventPolarity*
- *uint32\_t EventSensitivity*
- *uint32\_t EventFilter*

#### Field Documentation

- *uint32\_t HRTIM\_SimpleCaptureChannelCfgTypeDef::Event*  
Specifies the external event triggering the capture. This parameter can be any 'EEVx' value of [HRTIM\\_External\\_Event\\_Channels](#)
- *uint32\_t HRTIM\_SimpleCaptureChannelCfgTypeDef::EventPolarity*  
Specifies the polarity of the external event (in case of level sensitivity). This parameter can be a value of [HRTIM\\_External\\_Event\\_Polarity](#)
- *uint32\_t HRTIM\_SimpleCaptureChannelCfgTypeDef::EventSensitivity*  
Specifies the sensitivity of the external event. This parameter can be a value of [HRTIM\\_External\\_Event\\_Sensitivity](#)
- *uint32\_t HRTIM\_SimpleCaptureChannelCfgTypeDef::EventFilter*  
Defines the frequency used to sample the External Event and the length of the digital filter. This parameter can be a value of [HRTIM\\_External\\_Event\\_Filter](#)

### 23.1.8 HRTIM\_SimpleOnePulseChannelCfgTypeDef

#### Data Fields

- *uint32\_t Pulse*
- *uint32\_t OutputPolarity*
- *uint32\_t OutputIdleLevel*
- *uint32\_t Event*
- *uint32\_t EventPolarity*
- *uint32\_t EventSensitivity*
- *uint32\_t EventFilter*

#### Field Documentation

- *uint32\_t HRTIM\_SimpleOnePulseChannelCfgTypeDef::Pulse*  
Specifies the compare value to be loaded into the Compare Register. The compare value must be above or equal to 3 periods of the fHRTIM clock
- *uint32\_t HRTIM\_SimpleOnePulseChannelCfgTypeDef::OutputPolarity*  
Specifies the output polarity. This parameter can be any value of [HRTIM\\_Output\\_Polarity](#)

- ***uint32\_t HRTIM\_SimpleOnePulseChannelCfgTypeDef::OutputIdleLevel***  
Specifies whether the output level is active or inactive when in IDLE state. This parameter can be any value of [\*HRTIM\\_Output\\_IDLE\\_Level\*](#)
- ***uint32\_t HRTIM\_SimpleOnePulseChannelCfgTypeDef::Event***  
Specifies the external event triggering the pulse generation. This parameter can be any 'EEVx' value of [\*HRTIM\\_External\\_Event\\_Channels\*](#)
- ***uint32\_t HRTIM\_SimpleOnePulseChannelCfgTypeDef::EventPolarity***  
Specifies the polarity of the external event (in case of level sensitivity). This parameter can be a value of [\*HRTIM\\_External\\_Event\\_Polarity\*](#)
- ***uint32\_t HRTIM\_SimpleOnePulseChannelCfgTypeDef::EventSensitivity***  
Specifies the sensitivity of the external event. This parameter can be a value of [\*HRTIM\\_External\\_Event\\_Sensitivity\*](#).
- ***uint32\_t HRTIM\_SimpleOnePulseChannelCfgTypeDef::EventFilter***  
Defines the frequency used to sample the External Event and the length of the digital filter. This parameter can be a value of [\*HRTIM\\_External\\_Event\\_Filter\*](#)

### 23.1.9 HRTIM\_TimerCfgTypeDef

#### Data Fields

- ***uint32\_t InterruptRequests***
- ***uint32\_t DMARequests***
- ***uint32\_t DMASrcAddress***
- ***uint32\_t DMADstAddress***
- ***uint32\_t DMASize***
- ***uint32\_t HalfModeEnable***
- ***uint32\_t StartOnSync***
- ***uint32\_t ResetOnSync***
- ***uint32\_t DACSynchro***
- ***uint32\_t PreloadEnable***
- ***uint32\_t UpdateGating***
- ***uint32\_t BurstMode***
- ***uint32\_t RepetitionUpdate***
- ***uint32\_t PushPull***
- ***uint32\_t FaultEnable***
- ***uint32\_t FaultLock***
- ***uint32\_t DeadTimeInsertion***
- ***uint32\_t DelayedProtectionMode***
- ***uint32\_t UpdateTrigger***
- ***uint32\_t ResetTrigger***
- ***uint32\_t ResetUpdate***

#### Field Documentation

- ***uint32\_t HRTIM\_TimerCfgTypeDef::InterruptRequests***  
Relevant for all HRTIM timers, including the master. Specifies which interrupts requests must be enabled for the timer. This parameter can be any combination of [\*HRTIM\\_Master\\_Interrupt\\_Enable\*](#) or [\*HRTIM\\_Timing\\_Unit\\_Interrupt\\_Enable\*](#)
- ***uint32\_t HRTIM\_TimerCfgTypeDef::DMARequests***  
Relevant for all HRTIM timers, including the master. Specifies which DMA requests must be enabled for the timer. This parameter can be any combination of [\*HRTIM\\_Master\\_DMA\\_Request\\_Enable\*](#) or [\*HRTIM\\_Timing\\_Unit\\_DMA\\_Request\\_Enable\*](#)
- ***uint32\_t HRTIM\_TimerCfgTypeDef::DMASrcAddress***  
Relevant for all HRTIM timers, including the master. Specifies the address of the source address of the DMA transfer

- **`uint32_t HRTIM_TimerCfgTypeDef::DMADstAddress`**  
Relevant for all HRTIM timers, including the master. Specifies the address of the destination address of the DMA transfer
- **`uint32_t HRTIM_TimerCfgTypeDef::DMASize`**  
Relevant for all HRTIM timers, including the master. Specifies the size of the DMA transfer
- **`uint32_t HRTIM_TimerCfgTypeDef::HalfModeEnable`**  
Relevant for all HRTIM timers, including the master. Specifies whether or not half mode is enabled. This parameter can be any value of [`HRTIM\_Half\_Mode\_Enable`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::StartOnSync`**  
Relevant for all HRTIM timers, including the master. Specifies whether or not timer is reset by a rising edge on the synchronization input (when enabled). This parameter can be any value of [`HRTIM\_Start\_On\_Sync\_Input\_Event`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::ResetOnSync`**  
Relevant for all HRTIM timers, including the master. Specifies whether or not timer is reset by a rising edge on the synchronization input (when enabled). This parameter can be any value of [`HRTIM\_Reset\_On\_Sync\_Input\_Event`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::DACSynchro`**  
Relevant for all HRTIM timers, including the master. Indicates whether or not a DAC synchronization event is generated. This parameter can be any value of [`HRTIM\_DAC\_Synchronization`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::PreloadEnable`**  
Relevant for all HRTIM timers, including the master. Specifies whether or not register preload is enabled. This parameter can be any value of [`HRTIM\_Register\_Preload\_Enable`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::UpdateGating`**  
Relevant for all HRTIM timers, including the master. Specifies how the update occurs with respect to a burst DMA transaction or update enable inputs (Slave timers only). This parameter can be any value of [`HRTIM\_Update\_Gating`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::BurstMode`**  
Relevant for all HRTIM timers, including the master. Specifies how the timer behaves during a burst mode operation. This parameter can be any value of [`HRTIM\_Timer\_Burst\_Mode`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::RepetitionUpdate`**  
Relevant for all HRTIM timers, including the master. Specifies whether or not registers update is triggered by the repetition event. This parameter can be any value of [`HRTIM\_Timer\_Repetition\_Update`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::PushPull`**  
Relevant for Timer A to Timer E. Specifies whether or not the push-pull mode is enabled. This parameter can be any value of [`HRTIM\_Timer\_Push\_Pull\_Mode`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::FaultEnable`**  
Relevant for Timer A to Timer E. Specifies which fault channels are enabled for the timer. This parameter can be a combination of [`HRTIM\_Timer\_Fault\_Enabling`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::FaultLock`**  
Relevant for Timer A to Timer E. Specifies whether or not fault enabling status is write protected. This parameter can be a value of [`HRTIM\_Timer\_Fault\_Lock`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::DeadTimeInsertion`**  
Relevant for Timer A to Timer E. Specifies whether or not dead-time insertion is enabled for the timer. This parameter can be a value of [`HRTIM\_Timer\_Deadtime\_Insertion`](#)
- **`uint32_t HRTIM_TimerCfgTypeDef::DelayedProtectionMode`**  
Relevant for Timer A to Timer E. Specifies the delayed protection mode. This parameter can be a value of [`HRTIM\_Timer\_Delayed\_Protection\_Mode`](#)

- ***uint32\_t HRTIM\_TimerCfgTypeDef::UpdateTrigger***  
Relevant for Timer A to Timer E. Specifies source(s) triggering the timer registers update. This parameter can be a combination of [\*HRTIM\\_Timer\\_Update\\_Trigger\*](#)
- ***uint32\_t HRTIM\_TimerCfgTypeDef::ResetTrigger***  
Relevant for Timer A to Timer E. Specifies source(s) triggering the timer counter reset. This parameter can be a combination of [\*HRTIM\\_Timer\\_Reset\\_Trigger\*](#)
- ***uint32\_t HRTIM\_TimerCfgTypeDef::ResetUpdate***  
Relevant for Timer A to Timer E. Specifies whether or not registers update is triggered when the timer counter is reset. This parameter can be a value of [\*HRTIM\\_Timer\\_Reset\\_Update\*](#)

### 23.1.10 HRTIM\_CompareCfgTypeDef

#### Data Fields

- ***uint32\_t CompareValue***
- ***uint32\_t AutoDelayedMode***
- ***uint32\_t AutoDelayedTimeout***

#### Field Documentation

- ***uint32\_t HRTIM\_CompareCfgTypeDef::CompareValue***  
Specifies the compare value of the timer compare unit. The minimum value must be greater than or equal to 3 periods of the fHRTIM clock. The maximum value must be less than or equal to 0xFFFFFU - 1 periods of the fHRTIM clock
- ***uint32\_t HRTIM\_CompareCfgTypeDef::AutoDelayedMode***  
Specifies the auto delayed mode for compare unit 2 or 4. This parameter can be a value of [\*HRTIM\\_Compare\\_Unit\\_Auto\\_Delayed\\_Mode\*](#)
- ***uint32\_t HRTIM\_CompareCfgTypeDef::AutoDelayedTimeout***  
Specifies compare value for timing unit 1 or 3 when auto delayed mode with time out is selected. CompareValue + AutoDelayedTimeout must be less than 0xFFFFU

### 23.1.11 HRTIM\_CaptureCfgTypeDef

#### Data Fields

- ***uint32\_t Trigger***

#### Field Documentation

- ***uint32\_t HRTIM\_CaptureCfgTypeDef::Trigger***  
Specifies source(s) triggering the capture. This parameter can be a combination of [\*HRTIM\\_Capture\\_Unit\\_Trigger\*](#)

### 23.1.12 HRTIM\_OutputCfgTypeDef

#### Data Fields

- ***uint32\_t Polarity***
- ***uint32\_t SetSource***
- ***uint32\_t ResetSource***
- ***uint32\_t IdleMode***
- ***uint32\_t IdleLevel***
- ***uint32\_t FaultLevel***
- ***uint32\_t ChopperModeEnable***
- ***uint32\_t BurstModeEntryDelayed***

**Field Documentation**

- ***uint32\_t HRTIM\_OutputCfgTypeDef::Polarity***  
Specifies the output polarity. This parameter can be any value of [\*HRTIM\\_Output\\_Polarity\*](#)
- ***uint32\_t HRTIM\_OutputCfgTypeDef::SetSource***  
Specifies the event(s) transitioning the output from its inactive level to its active level. This parameter can be a combination of [\*HRTIM\\_Output\\_Set\\_Source\*](#)
- ***uint32\_t HRTIM\_OutputCfgTypeDef::ResetSource***  
Specifies the event(s) transitioning the output from its active level to its inactive level. This parameter can be a combination of [\*HRTIM\\_Output\\_Reset\\_Source\*](#)
- ***uint32\_t HRTIM\_OutputCfgTypeDef::IdleMode***  
Specifies whether or not the output is affected by a burst mode operation. This parameter can be any value of [\*HRTIM\\_Output\\_Idle\\_Mode\*](#)
- ***uint32\_t HRTIM\_OutputCfgTypeDef::IdleLevel***  
Specifies whether the output level is active or inactive when in IDLE state. This parameter can be any value of [\*HRTIM\\_Output\\_IDLE\\_Level\*](#)
- ***uint32\_t HRTIM\_OutputCfgTypeDef::FaultLevel***  
Specifies whether the output level is active or inactive when in FAULT state. This parameter can be any value of [\*HRTIM\\_Output\\_FAULT\\_Level\*](#)
- ***uint32\_t HRTIM\_OutputCfgTypeDef::ChopperModeEnable***  
Indicates whether or not the chopper mode is enabled. This parameter can be any value of [\*HRTIM\\_Output\\_Chopper\\_Mode\\_Enable\*](#)
- ***uint32\_t HRTIM\_OutputCfgTypeDef::BurstModeEntryDelayed***  
Indicates whether or not dead-time is inserted when entering the IDLE state during a burst mode operation. This parameter can be any value of [\*HRTIM\\_Output\\_Burst\\_Mode\\_Entry\\_Delayed\*](#)

**23.1.13 HRTIM\_TimerEventFilteringCfgTypeDef****Data Fields**

- ***uint32\_t Filter***
- ***uint32\_t Latch***

**Field Documentation**

- ***uint32\_t HRTIM\_TimerEventFilteringCfgTypeDef::Filter***  
Specifies the type of event filtering within the timing unit. This parameter can be a value of [\*HRTIM\\_Timer\\_External\\_Event\\_Filter\*](#)
- ***uint32\_t HRTIM\_TimerEventFilteringCfgTypeDef::Latch***  
Specifies whether or not the signal is latched. This parameter can be a value of [\*HRTIM\\_Timer\\_External\\_Event\\_Latch\*](#)

**23.1.14 HRTIM\_DeadTimeCfgTypeDef****Data Fields**

- ***uint32\_t Prescaler***
- ***uint32\_t RisingValue***
- ***uint32\_t RisingSign***
- ***uint32\_t RisingLock***
- ***uint32\_t RisingSignLock***
- ***uint32\_t FallingValue***
- ***uint32\_t FallingSign***
- ***uint32\_t FallingLock***
- ***uint32\_t FallingSignLock***

**Field Documentation**

- ***uint32\_t HRTIM\_DeadTimeCfgTypeDef::Prescaler***  
Specifies the Deadtime Prescaler. This parameter can be a value of [\*\*HRTIM\\_Deadtime\\_Prescaler\\_Ratio\*\*](#)
- ***uint32\_t HRTIM\_DeadTimeCfgTypeDef::RisingValue***  
Specifies the Deadtime following a rising edge. This parameter can be a number between 0x0 and 0x1FFU
- ***uint32\_t HRTIM\_DeadTimeCfgTypeDef::RisingSign***  
Specifies whether the deadtime is positive or negative on rising edge. This parameter can be a value of [\*\*HRTIM\\_Deadtime\\_Rising\\_Sign\*\*](#)
- ***uint32\_t HRTIM\_DeadTimeCfgTypeDef::RisingLock***  
Specifies whether or not deadtime rising settings (value and sign) are write protected. This parameter can be a value of [\*\*HRTIM\\_Deadtime\\_Rising\\_Lock\*\*](#)
- ***uint32\_t HRTIM\_DeadTimeCfgTypeDef::RisingSignLock***  
Specifies whether or not deadtime rising sign is write protected. This parameter can be a value of [\*\*HRTIM\\_Deadtime\\_Rising\\_Sign\\_Lock\*\*](#)
- ***uint32\_t HRTIM\_DeadTimeCfgTypeDef::FallingValue***  
Specifies the Deadtime following a falling edge. This parameter can be a number between 0x0 and 0x1FFU
- ***uint32\_t HRTIM\_DeadTimeCfgTypeDef::FallingSign***  
Specifies whether the deadtime is positive or negative on falling edge. This parameter can be a value of [\*\*HRTIM\\_Deadtime\\_Falling\\_Sign\*\*](#)
- ***uint32\_t HRTIM\_DeadTimeCfgTypeDef::FallingLock***  
Specifies whether or not deadtime falling settings (value and sign) are write protected. This parameter can be a value of [\*\*HRTIM\\_Deadtime\\_Falling\\_Lock\*\*](#)
- ***uint32\_t HRTIM\_DeadTimeCfgTypeDef::FallingSignLock***  
Specifies whether or not deadtime falling sign is write protected. This parameter can be a value of [\*\*HRTIM\\_Deadtime\\_Falling\\_Sign\\_Lock\*\*](#)

**23.1.15 HRTIM\_ChopperModeCfgTypeDef****Data Fields**

- ***uint32\_t CarrierFreq***
- ***uint32\_t DutyCycle***
- ***uint32\_t StartPulse***

**Field Documentation**

- ***uint32\_t HRTIM\_ChopperModeCfgTypeDef::CarrierFreq***  
Specifies the Timer carrier frequency value. This parameter can be a value of [\*\*HRTIM\\_Chopper\\_Frequency\*\*](#)
- ***uint32\_t HRTIM\_ChopperModeCfgTypeDef::DutyCycle***  
Specifies the Timer chopper duty cycle value. This parameter can be a value of [\*\*HRTIM\\_Chopper\\_Duty\\_Cycle\*\*](#)
- ***uint32\_t HRTIM\_ChopperModeCfgTypeDef::StartPulse***  
Specifies the Timer pulse width value. This parameter can be a value of [\*\*HRTIM\\_Chopper\\_Start\\_Pulse\\_Width\*\*](#)

**23.1.16 HRTIM\_EventCfgTypeDef****Data Fields**

- ***uint32\_t Source***
- ***uint32\_t Polarity***
- ***uint32\_t Sensitivity***
- ***uint32\_t Filter***

- *uint32\_t FastMode*

#### Field Documentation

- *uint32\_t HRTIM\_EventCfgTypeDef::Source*

Identifies the source of the external event. This parameter can be a value of [\*HRTIM\\_External\\_Event\\_Sources\*](#)

- *uint32\_t HRTIM\_EventCfgTypeDef::Polarity*

Specifies the polarity of the external event (in case of level sensitivity). This parameter can be a value of [\*HRTIM\\_External\\_Event\\_Polarity\*](#)

- *uint32\_t HRTIM\_EventCfgTypeDef::Sensitivity*

Specifies the sensitivity of the external event. This parameter can be a value of [\*HRTIM\\_External\\_Event\\_Sensitivity\*](#)

- *uint32\_t HRTIM\_EventCfgTypeDef::Filter*

Defines the frequency used to sample the External Event and the length of the digital filter. This parameter can be a value of [\*HRTIM\\_External\\_Event\\_Filter\*](#)

- *uint32\_t HRTIM\_EventCfgTypeDef::FastMode*

Indicates whether or not low latency mode is enabled for the external event. This parameter can be a value of [\*HRTIM\\_External\\_Event\\_Fast\\_Mode\*](#)

### 23.1.17 HRTIM\_FaultCfgTypeDef

#### Data Fields

- *uint32\_t Source*
- *uint32\_t Polarity*
- *uint32\_t Filter*
- *uint32\_t Lock*

#### Field Documentation

- *uint32\_t HRTIM\_FaultCfgTypeDef::Source*

Identifies the source of the fault. This parameter can be a value of [\*HRTIM\\_Fault\\_Sources\*](#)

- *uint32\_t HRTIM\_FaultCfgTypeDef::Polarity*

Specifies the polarity of the fault event. This parameter can be a value of [\*HRTIM\\_Fault\\_Polarity\*](#)

- *uint32\_t HRTIM\_FaultCfgTypeDef::Filter*

Defines the frequency used to sample the Fault input and the length of the digital filter. This parameter can be a value of [\*HRTIM\\_Fault\\_Filter\*](#)

- *uint32\_t HRTIM\_FaultCfgTypeDef::Lock*

Indicates whether or not fault programming bits are write protected. This parameter can be a value of [\*HRTIM\\_Fault\\_Lock\*](#)

### 23.1.18 HRTIM\_BurstModeCfgTypeDef

#### Data Fields

- *uint32\_t Mode*
- *uint32\_t ClockSource*
- *uint32\_t Prescaler*
- *uint32\_t PreloadEnable*
- *uint32\_t Trigger*
- *uint32\_t IdleDuration*
- *uint32\_t Period*

**Field Documentation**

- ***uint32\_t HRTIM\_BurstModeCfgTypeDef::Mode***  
Specifies the burst mode operating mode. This parameter can be a value of [\*\*HRTIM\\_Burst\\_Mode\\_Operating\\_Mode\*\*](#)
- ***uint32\_t HRTIM\_BurstModeCfgTypeDef::ClockSource***  
Specifies the burst mode clock source. This parameter can be a value of [\*\*HRTIM\\_Burst\\_Mode\\_Clock\\_Source\*\*](#)
- ***uint32\_t HRTIM\_BurstModeCfgTypeDef::Prescaler***  
Specifies the burst mode prescaler. This parameter can be a value of [\*\*HRTIM\\_Burst\\_Mode\\_Prescaler\*\*](#)
- ***uint32\_t HRTIM\_BurstModeCfgTypeDef::PreloadEnable***  
Specifies whether or not preload is enabled for burst mode related registers (HRTIM\_BMCMPR and HRTIM\_BMPER). This parameter can be a combination of [\*\*HRTIM\\_Burst\\_Mode\\_Register\\_Preload\\_Enable\*\*](#)
- ***uint32\_t HRTIM\_BurstModeCfgTypeDef::Trigger***  
Specifies the event(s) triggering the burst operation. This parameter can be a combination of [\*\*HRTIM\\_Burst\\_Mode\\_Trigger\*\*](#)
- ***uint32\_t HRTIM\_BurstModeCfgTypeDef::IdleDuration***  
Specifies number of periods during which the selected timers are in idle state. This parameter can be a number between 0x0 and 0xFFFF
- ***uint32\_t HRTIM\_BurstModeCfgTypeDef::Period***  
Specifies burst mode repetition period. This parameter can be a number between 0x1 and 0xFFFF

**23.1.19 HRTIM\_ADCTriggerCfgTypeDef****Data Fields**

- ***uint32\_t UpdateSource***
- ***uint32\_t Trigger***

**Field Documentation**

- ***uint32\_t HRTIM\_ADCTriggerCfgTypeDef::UpdateSource***  
Specifies the ADC trigger update source. This parameter can be a combination of [\*\*HRTIM\\_ADC\\_Trigger\\_Update\\_Source\*\*](#)
- ***uint32\_t HRTIM\_ADCTriggerCfgTypeDef::Trigger***  
Specifies the event(s) triggering the ADC conversion. This parameter can be a value of [\*\*HRTIM\\_ADC\\_Trigger\\_Event\*\*](#)

**23.2 HRTIM Firmware driver API description****23.2.1 Simple mode v.s. waveform mode**

The HRTIM HAL API is split into 2 categories:

1. Simple functions: these functions allow for using a HRTIM timer as a general purpose timer with high resolution capabilities. HRTIM simple modes are managed through the set of functions named `HAL_HRTIM_Simple<Function>`. These functions are similar in name and usage to the one defined for the TIM peripheral. When a HRTIM timer operates in simple mode, only a very limited set of HRTIM features are used.  
Following simple modes are proposed:
  - Output compare mode,
  - PWM output mode,
  - Input capture mode,
  - One pulse mode.

2. Waveform functions: These functions allow taking advantage of the HRTIM flexibility to produce numerous types of control signal. When a HRTIM timer operates in waveform mode, all the HRTIM features are accessible without any restriction. HRTIM waveform modes are managed through the set of functions named `HAL_HRTIM_Waveform<Function>`

### 23.2.2 How to use this driver

1. Initialize the HRTIM low level resources by implementing the `HAL_HRTIM_MsplInit()` function:
  - a. Enable the HRTIM clock source using `__HRTIMx_CLK_ENABLE()`
  - b. Connect HRTIM pins to MCU I/Os
    - Enable the clock for the HRTIM GPIOs using the following function:  
`__HAL_RCC_GPIOx_CLK_ENABLE()`
    - Configure these GPIO pins in Alternate Function mode using  
`HAL_GPIO_Init()`
  - c. When using DMA to control data transfer (e.g `HAL_HRTIM_SimpleBaseStart_DMA()`)
    - Enable the DMAx interface clock using `__DMAx_CLK_ENABLE()`
    - Initialize the DMA handle
    - Associate the initialized DMA handle to the appropriate DMA handle of the HRTIM handle using `__HAL_LINKDMA()`
    - Initialize the DMA channel using `HAL_DMA_Init()`
    - Configure the priority and enable the NVIC for the transfer complete interrupt on the DMA channel using `HAL_NVIC_SetPriority()` and `HAL_NVIC_EnableIRQ()`
  - d. In case of using interrupt mode (e.g `HAL_HRTIM_SimpleBaseStart_IT()`)
    - Configure the priority and enable the NVIC for the concerned HRTIM interrupt using `HAL_NVIC_SetPriority()` and `HAL_NVIC_EnableIRQ()`
2. Initialize the HRTIM HAL using `HAL_HRTIM_Init()`. The HRTIM configuration structure (field of the HRTIM handle) specifies which global interrupt of whole HRTIM must be enabled (Burst mode period, System fault, Faults). It also contains the HRTIM external synchronization configuration. HRTIM can act as a master (generating a synchronization signal) or as a slave (waiting for a trigger to be synchronized).
3. Start the high resolution unit using `HAL_HRTIM_DLLCalibrationStart()`. DLL calibration is executed periodically and compensate for potential voltage and temperature drifts. DLL calibration period is specified by the `CalibrationRate` argument.
4. HRTIM timers cannot be used until the high resolution unit is ready. This can be checked using `HAL_HRTIM_PollForDLLCalibration()`: this function returns `HAL_OK` if DLL calibration is completed or `HAL_TIMEOUT` if the DLL calibration is still going on when timeout given as argument expires. DLL calibration can also be started in interrupt mode using `HAL_HRTIM_DLLCalibrationStart_IT()`. In that case an interrupt is generated when the DLL calibration is completed. Note that as DLL calibration is executed on a periodic basis an interrupt will be generated at the end of every DLL calibration operation (worst case: one interrupt every 14 micro seconds !).
5. Configure HRTIM resources shared by all HRTIM timers
  - a. Burst Mode Controller:
    - `HAL_HRTIM_BurstModeConfig()`: configures the HRTIM burst mode controller: operating mode (continuous or one-shot mode), clock (source, prescaler) , trigger(s), period, idle duration.
  - b. External Events Conditionning:
    - `HAL_HRTIM_EventConfig()`: configures the conditioning of an external event channel: source, polarity, edge-sensitivity. External event can be used as triggers (timer reset, input capture, burst mode, ADC triggers, delayed

- protection) They can also be used to set or reset timer outputs. Up to 10 event channels are available.
- HAL\_HRTIM\_EventPrescalerConfig(): configures the external event sampling clock (used for digital filtering).
- c. Fault Conditionning:
- HAL\_HRTIM\_FaultConfig(): configures the conditioning of a fault channel: source, polarity, edge-sensitivity. Fault channels are used to disable the outputs in case of an abnormal operation. Up to 5 fault channels are available.
  - HAL\_HRTIM\_FaultPrescalerConfig(): configures the fault sampling clock (used for digital filtering).
  - HAL\_HRTIM\_FaultModeCtl(): Enables or disables fault input(s) circuitry. By default all fault inputs are disabled.
- d. ADC trigger:
- HAL\_HRTIM\_ADCTriggerConfig(): configures the source triggering the update of the ADC trigger register and the ADC trigger. 4 independent triggers are available to start both the regular and the injected sequencers of the 2 ADCs
6. Configure HRTIM timer time base using HAL\_HRTIM\_TimeBaseConfig(). This function must be called whatever the HRTIM timer operating mode is (simple v.s. waveform). It configures mainly:
- a. The HRTIM timer counter operating mode (continuous v.s. one shot)
  - b. The HRTIM timer clock prescaler
  - c. The HRTIM timer period
  - d. The HRTIM timer repetition counter

### If the HRTIM timer operates in simple mode

1. Start or Stop simple timers
  - Simple time base:  
HAL\_HRTIM\_SimpleBaseStart(), HAL\_HRTIM\_SimpleBaseStop(),  
HAL\_HRTIM\_SimpleBaseStart\_IT(), HAL\_HRTIM\_SimpleBaseStop\_IT(),  
HAL\_HRTIM\_SimpleBaseStart\_DMA(), HAL\_HRTIM\_SimpleBaseStop\_DMA().
  - Simple output compare: HAL\_HRTIM\_SimpleOCChannelConfig(),  
HAL\_HRTIM\_SimpleOCStart(), HAL\_HRTIM\_SimpleOCStop(),  
HAL\_HRTIM\_SimpleOCStart\_IT(), HAL\_HRTIM\_SimpleOCStop\_IT(),  
HAL\_HRTIM\_SimpleOCStart\_DMA(), HAL\_HRTIM\_SimpleOCStop\_DMA(),
  - Simple PWM output: HAL\_HRTIM\_SimplePWMChannelConfig(),  
HAL\_HRTIM\_SimplePWMStart(), HAL\_HRTIM\_SimplePWMStop(),  
HAL\_HRTIM\_SimplePWMStart\_IT(), HAL\_HRTIM\_SimplePWMStop\_IT(),  
HAL\_HRTIM\_SimplePWMStart\_DMA(), HAL\_HRTIM\_SimplePWMStop\_DMA(),
  - Simple input capture: HAL\_HRTIM\_SimpleCaptureChannelConfig(),  
HAL\_HRTIM\_SimpleCaptureStart(), HAL\_HRTIM\_SimpleCaptureStop(),  
HAL\_HRTIM\_SimpleCaptureStart\_IT(), HAL\_HRTIM\_SimpleCaptureStop\_IT(),  
HAL\_HRTIM\_SimpleCaptureStart\_DMA(), HAL\_HRTIM\_SimpleCaptureStop\_DM\_A().
  - Simple one pulse: HAL\_HRTIM\_SimpleOnePulseChannelConfig(),  
HAL\_HRTIM\_SimpleOnePulseStart(), HAL\_HRTIM\_SimpleOnePulseStop(),  
HAL\_HRTIM\_SimpleOnePulseStart\_IT(), HAL\_HRTIM\_SimpleOnePulseStop\_IT().

### If the HRTIM timer operates in waveform mode

1. Completes waveform timer configuration
  - HAL\_HRTIM\_WaveformTimerConfig(): configuration of a HRTIM timer operating in wave form mode mainly consists in:

- Enabling the HRTIM timer interrupts and DMA requests.
  - Enabling the half mode for the HRTIM timer.
  - Defining how the HRTIM timer reacts to external synchronization input.
  - Enabling the push-pull mode for the HRTIM timer.
  - Enabling the fault channels for the HRTIM timer.
  - Enabling the dead-time insertion for the HRTIM timer.
  - Setting the delayed protection mode for the HRTIM timer (source and outputs on which the delayed protection are applied).
  - Specifying the HRTIM timer update and reset triggers.
  - Specifying the HRTIM timer registers update policy (e.g. pre-load enabling).
  - HAL\_HRTIM\_TimerEventFilteringConfig(): configures external event blanking and windowing circuitry of a HRTIM timer:
    - Blanking: to mask external events during a defined time period a defined time period
    - Windowing, to enable external events only during a defined time period
  - HAL\_HRTIM\_DeadTimeConfig(): configures the dead-time insertion unit for a HRTIM timer. Allows to generate a couple of complementary signals from a single reference waveform, with programmable delays between active state.
  - HAL\_HRTIM\_ChopperModeConfig(): configures the parameters of the high-frequency carrier signal added on top of the timing unit output. Chopper mode can be enabled or disabled for each timer output separately (see HAL\_HRTIM\_WaveformOutputConfig()).
  - HAL\_HRTIM\_BurstDMAConfig(): configures the burst DMA burst controller. Allows having multiple HRTIM registers updated with a single DMA request. The burst DMA operation is started by calling HAL\_HRTIM\_BurstDMATransfer().
  - HAL\_HRTIM\_WaveformCompareConfig(): configures the compare unit of a HRTIM timer. This operation consists in setting the compare value and possibly specifying the auto delayed mode for compare units 2 and 4 (allows to have compare events generated relatively to capture events). Note that when auto delayed mode is needed, the capture unit associated to the compare unit must be configured separately.
  - HAL\_HRTIM\_WaveformCaptureConfig(): configures the capture unit of a HRTIM timer. This operation consists in specifying the source(s) triggering the capture (timer register update event, external event, timer output set/reset event, other HRTIM timer related events).
  - HAL\_HRTIM\_WaveformOutputConfig(): configuration of a HRTIM timer output mainly consists in:
    - Setting the output polarity (active high or active low),
    - Defining the set/reset crossbar for the output,
    - Specifying the fault level (active or inactive) in IDLE and FAULT states.,
2. Set waveform timer output(s) level
  - HAL\_HRTIM\_WaveformSetOutputLevel(): forces the output to its active or inactive level. For example, when deadtime insertion is enabled it is necessary to force the output level by software to have the outputs in a complementary state as soon as the RUN mode is entered.
3. Enable or Disable waveform timer output(s)
  - HAL\_HRTIM\_WaveformOutputStart(),HAL\_HRTIM\_WaveformOutputStop().
4. Start or Stop waveform HRTIM timer(s).
  - HAL\_HRTIM\_WaveformCounterStart(),HAL\_HRTIM\_WaveformCounterStop(),
  - HAL\_HRTIM\_WaveformCounterStart\_IT(),HAL\_HRTIM\_WaveformCounterStop\_IT(),
  - HAL\_HRTIM\_WaveformCounterStart()\_DMA,HAL\_HRTIM\_WaveformCounterStop\_DMA(),

5. Burst mode controller enabling:
  - HAL\_HRTIM\_BurstModeCtl(): activates or de-activates the burst mode controller.
6. Some HRTIM operations can be triggered by software:
  - HAL\_HRTIM\_BurstModeSoftwareTrigger(): calling this function trigs the burst operation.
  - HAL\_HRTIM\_SoftwareCapture(): calling this function trigs the capture of the HRTIM timer counter.
  - HAL\_HRTIM\_SoftwareUpdate(): calling this function trigs the update of the pre-loadable registers of the HRTIM timer
  - HAL\_HRTIM\_SoftwareReset(): calling this function resets the HRTIM timer counter.
7. Some functions can be used any time to retrieve HRTIM timer related information
  - HAL\_HRTIM\_GetCapturedValue(): returns actual value of the capture register of the designated capture unit.
  - HAL\_HRTIM\_WaveformGetOutputLevel(): returns actual level (ACTIVE/INACTIVE) of the designated timer output.
  - HAL\_HRTIM\_WaveformGetOutputState(): returns actual state (IDLE/RUN/FAULT) of the designated timer output.
  - HAL\_HRTIM\_GetDelayedProtectionStatus(): returns actual level (ACTIVE/INACTIVE) of the designated output when the delayed protection was triggered.
  - HAL\_HRTIM\_GetBurstStatus(): returns the actual status (ACTIVE/INACTIVE) of the burst mode controller.
  - HAL\_HRTIM\_GetCurrentPushPullStatus(): when the push-pull mode is enabled for the HRTIM timer (see HAL\_HRTIM\_WaveformTimerConfig()), the push-pull status indicates on which output the signal is currently active (e.g signal applied on output 1 and output 2 forced inactive or vice versa).
  - HAL\_HRTIM\_GetIdlePushPullStatus(): when the push-pull mode is enabled for the HRTIM timer (see HAL\_HRTIM\_WaveformTimerConfig()), the idle push-pull status indicates during which period the delayed protection request occurred (e.g. protection occurred when the output 1 was active and output 2 forced inactive or vice versa).
8. Some functions can be used any time to retrieve actual HRTIM status
  - HAL\_HRTIM\_GetState(): returns actual HRTIM instance HAL state.

### 23.2.3 Initialization and Time Base Configuration functions

This section provides functions allowing to:

- Initialize a HRTIM instance
- De-initialize a HRTIM instance
- Initialize the HRTIM MSP
- De-initialize the HRTIM MSP
- Start the high-resolution unit (start DLL calibration)
- Check that the high resolution unit is ready (DLL calibration done)
- Configure the time base unit of a HRTIM timer

This section contains the following APIs:

- [\*\*HAL\\_HRTIM\\_Init\(\)\*\*](#)
- [\*\*HAL\\_HRTIM\\_DelInit\(\)\*\*](#)
- [\*\*HAL\\_HRTIM\\_MspInit\(\)\*\*](#)
- [\*\*HAL\\_HRTIM\\_MspDelInit\(\)\*\*](#)
- [\*\*HAL\\_HRTIM\\_DLLCalibrationStart\(\)\*\*](#)

- [`HAL\_HRTIM\_DLLCalibrationStart\_IT\(\)`](#)
- [`HAL\_HRTIM\_PollForDLLCalibration\(\)`](#)
- [`HAL\_HRTIM\_TimeBaseConfig\(\)`](#)

### 23.2.4 Simple time base mode functions

This section provides functions allowing to:

- Start simple time base
- Stop simple time base
- Start simple time base and enable interrupt
- Stop simple time base and disable interrupt
- Start simple time base and enable DMA transfer
- Stop simple time base and disable DMA transfer When a HRTIM timer operates in simple time base mode, the timer counter counts from 0 to the period value.

This section contains the following APIs:

- [`HAL\_HRTIM\_SimpleBaseStart\(\)`](#)
- [`HAL\_HRTIM\_SimpleBaseStop\(\)`](#)
- [`HAL\_HRTIM\_SimpleBaseStart\_IT\(\)`](#)
- [`HAL\_HRTIM\_SimpleBaseStop\_IT\(\)`](#)
- [`HAL\_HRTIM\_SimpleBaseStart\_DMA\(\)`](#)
- [`HAL\_HRTIM\_SimpleBaseStop\_DMA\(\)`](#)

### 23.2.5 Simple output compare functions

This section provides functions allowing to:

- Configure simple output channel
- Start simple output compare
- Stop simple output compare
- Start simple output compare and enable interrupt
- Stop simple output compare and disable interrupt
- Start simple output compare and enable DMA transfer
- Stop simple output compare and disable DMA transfer When a HRTIM timer operates in simple output compare mode the output level is set to a programmable value when a match is found between the compare register and the counter. Compare unit 1 is automatically associated to output 1 Compare unit 2 is automatically associated to output 2

This section contains the following APIs:

- [`HAL\_HRTIM\_SimpleOCChannelConfig\(\)`](#)
- [`HAL\_HRTIM\_SimpleOCStart\(\)`](#)
- [`HAL\_HRTIM\_SimpleOCStop\(\)`](#)
- [`HAL\_HRTIM\_SimpleOCStart\_IT\(\)`](#)
- [`HAL\_HRTIM\_SimpleOCStop\_IT\(\)`](#)
- [`HAL\_HRTIM\_SimpleOCStart\_DMA\(\)`](#)
- [`HAL\_HRTIM\_SimpleOCStop\_DMA\(\)`](#)

### 23.2.6 Simple PWM output functions

This section provides functions allowing to:

- Configure simple PWM output channel
- Start simple PWM output
- Stop simple PWM output

- Start simple PWM output and enable interrupt
- Stop simple PWM output and disable interrupt
- Start simple PWM output and enable DMA transfer
- Stop simple PWM output and disable DMA transfer When a HRTIM timer operates in simple PWM output mode the output level is set to a programmable value when a match is found between the compare register and the counter and reset when the timer period is reached. Duty cycle is determined by the comparison value. Compare unit 1 is automatically associated to output 1 Compare unit 2 is automatically associated to output 2

This section contains the following APIs:

- [`HAL\_HRTIM\_SimplePWMChannelConfig\(\)`](#)
- [`HAL\_HRTIM\_SimplePWMStart\(\)`](#)
- [`HAL\_HRTIM\_SimplePWMStop\(\)`](#)
- [`HAL\_HRTIM\_SimplePWMStart\_IT\(\)`](#)
- [`HAL\_HRTIM\_SimplePWMStop\_IT\(\)`](#)
- [`HAL\_HRTIM\_SimplePWMStart\_DMA\(\)`](#)
- [`HAL\_HRTIM\_SimplePWMStop\_DMA\(\)`](#)

### 23.2.7 Simple input capture functions

This section provides functions allowing to:

- Configure simple input capture channel
- Start simple input capture
- Stop simple input capture
- Start simple input capture and enable interrupt
- Stop simple input capture and disable interrupt
- Start simple input capture and enable DMA transfer
- Stop simple input capture and disable DMA transfer When a HRTIM timer operates in simple input capture mode the Capture Register (HRTIM\_CPT1/2xR) is used to latch the value of the timer counter counter after a transition detected on a given external event input.

This section contains the following APIs:

- [`HAL\_HRTIM\_SimpleCaptureChannelConfig\(\)`](#)
- [`HAL\_HRTIM\_SimpleCaptureStart\(\)`](#)
- [`HAL\_HRTIM\_SimpleCaptureStop\(\)`](#)
- [`HAL\_HRTIM\_SimpleCaptureStart\_IT\(\)`](#)
- [`HAL\_HRTIM\_SimpleCaptureStop\_IT\(\)`](#)
- [`HAL\_HRTIM\_SimpleCaptureStart\_DMA\(\)`](#)
- [`HAL\_HRTIM\_SimpleCaptureStop\_DMA\(\)`](#)

### 23.2.8 Simple one pulse functions

This section provides functions allowing to:

- Configure one pulse channel
- Start one pulse generation
- Stop one pulse generation
- Start one pulse generation and enable interrupt
- Stop one pulse generation and disable interrupt When a HRTIM timer operates in simple one pulse mode the timer counter is started in response to transition detected on a given external event input to generate a pulse with a programmable length after a programmable delay.

This section contains the following APIs:

- [\*HAL\\_HRTIM\\_SimpleOnePulseChannelConfig\(\)\*](#)
- [\*HAL\\_HRTIM\\_SimpleOnePulseStart\(\)\*](#)
- [\*HAL\\_HRTIM\\_SimpleOnePulseStop\(\)\*](#)
- [\*HAL\\_HRTIM\\_SimpleOnePulseStart\\_IT\(\)\*](#)
- [\*HAL\\_HRTIM\\_SimpleOnePulseStop\\_IT\(\)\*](#)

### 23.2.9 HRTIM configuration functions

This section provides functions allowing to configure the HRTIM resources shared by all the HRTIM timers operating in waveform mode:

- Configure the burst mode controller
- Configure an external event conditionning
- Configure the external events sampling clock
- Configure a fault conditionning
- Enable or disable fault inputs
- Configure the faults sampling clock
- Configure an ADC trigger

This section contains the following APIs:

- [\*HAL\\_HRTIM\\_BurstModeConfig\(\)\*](#)
- [\*HAL\\_HRTIM\\_EventConfig\(\)\*](#)
- [\*HAL\\_HRTIM\\_EventPrescalerConfig\(\)\*](#)
- [\*HAL\\_HRTIM\\_FaultConfig\(\)\*](#)
- [\*HAL\\_HRTIM\\_FaultPrescalerConfig\(\)\*](#)
- [\*HAL\\_HRTIM\\_FaultModeCtl\(\)\*](#)
- [\*HAL\\_HRTIM\\_ADCTriggerConfig\(\)\*](#)

### 23.2.10 HRTIM timer configuration and control functions

This section provides functions used to configure and control a HRTIM timer operating in waveform mode:

- Configure HRTIM timer general behavior
- Configure HRTIM timer event filtering
- Configure HRTIM timer deadtime insertion
- Configure HRTIM timer chopper mode
- Configure HRTIM timer burst DMA
- Configure HRTIM timer compare unit
- Configure HRTIM timer capture unit
- Configure HRTIM timer output
- Set HRTIM timer output level
- Enable HRTIM timer output
- Disable HRTIM timer output
- Start HRTIM timer
- Stop HRTIM timer
- Start HRTIM timer and enable interrupt
- Stop HRTIM timer and disable interrupt
- Start HRTIM timer and enable DMA transfer
- Stop HRTIM timer and disable DMA transfer
- Enable or disable the burst mode controller
- Start the burst mode controller (by software)
- Trigger a Capture (by software)

- Update the HRTIM timer preloadable registers (by software)
- Reset the HRTIM timer counter (by software)
- Start a burst DMA transfer
- Enable timer register update
- Disable timer register update

This section contains the following APIs:

- `HAL_HRTIM_WaveformTimerConfig()`
- `HAL_HRTIM_TimerEventFilteringConfig()`
- `HAL_HRTIM_DeadTimeConfig()`
- `HAL_HRTIM_ChopperModeConfig()`
- `HAL_HRTIM_BurstDMAConfig()`
- `HAL_HRTIM_WaveformCompareConfig()`
- `HAL_HRTIM_WaveformCaptureConfig()`
- `HAL_HRTIM_WaveformOutputConfig()`
- `HAL_HRTIM_WaveformSetOutputLevel()`
- `HAL_HRTIM_WaveformOutputStart()`
- `HAL_HRTIM_WaveformOutputStop()`
- `HAL_HRTIM_WaveformCounterStart()`
- `HAL_HRTIM_WaveformCounterStop()`
- `HAL_HRTIM_WaveformCounterStart_IT()`
- `HAL_HRTIM_WaveformCounterStop_IT()`
- `HAL_HRTIM_WaveformCounterStart_DMA()`
- `HAL_HRTIM_WaveformCounterStop_DMA()`
- `HAL_HRTIM_BurstModeCtl()`
- `HAL_HRTIM_BurstModeSoftwareTrigger()`
- `HAL_HRTIM_SoftwareCapture()`
- `HAL_HRTIM_SoftwareUpdate()`
- `HAL_HRTIM_SoftwareReset()`
- `HAL_HRTIM_BurstDMATransfer()`
- `HAL_HRTIM_UpdateEnable()`
- `HAL_HRTIM_UpdateDisable()`

### 23.2.11 Peripheral State functions

This section provides functions used to get HRTIM or HRTIM timer specific information:

- Get HRTIM HAL state
- Get captured value
- Get HRTIM timer output level
- Get HRTIM timer output state
- Get delayed protection status
- Get burst status
- Get current push-pull status
- Get idle push-pull status

This section contains the following APIs:

- `HAL_HRTIM_GetState()`
- `HAL_HRTIM_GetCapturedValue()`
- `HAL_HRTIM_WaveformGetOutputLevel()`
- `HAL_HRTIM_WaveformGetOutputState()`
- `HAL_HRTIM_GetDelayedProtectionStatus()`
- `HAL_HRTIM_GetBurstStatus()`
- `HAL_HRTIM_GetCurrentPushPullStatus()`

- [\*\*\*HAL\\_HRTIM\\_GetIdlePushPullStatus\(\)\*\*\*](#)

### 23.2.12 Detailed description of functions

#### **HAL\_HRTIM\_Init**

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_Init (HRTIM_HandleTypeDefDef * hhrtim)</b>                      |
| Function description | Initializes a HRTIM instance.                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                         |

#### **HAL\_HRTIM\_DelInit**

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_DelInit (HRTIM_HandleTypeDefDef * hhrtim)</b>                   |
| Function description | De-initializes a HRTIM instance.                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                         |

#### **HAL\_HRTIM\_MspInit**

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_MspInit (HRTIM_HandleTypeDefDef * hhrtim)</b>                                |
| Function description | MSP initialization for a HRTIM instance.                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                |

#### **HAL\_HRTIM\_MspDelInit**

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_MspDelInit (HRTIM_HandleTypeDefDef * hhrtim)</b>                             |
| Function description | MSP de-initialization for a HRTIM instance.                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                |

#### **HAL\_HRTIM\_TimeBaseConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_TimeBaseConfig (HRTIM_HandleTypeDefDef * hhrtim, uint32_t TimerIdx, HRTIM_TimeBaseCfgTypeDef * pTimeBaseCfg)</b>                                                                                                                                                                                                                                                                                         |
| Function description | Configures the time base unit of a timer.                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_MASTER for master timer</li> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> </ul> </li> </ul> |

|               |                                                                                                                                                                                                                                                                                                                 |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul>                                                                                                                                                                        |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                          |
| Notes         | <ul style="list-style-type: none"> <li>• This function must be called prior starting the timer</li> <li>• The time-base unit initialization parameters specify: The timer counter operating mode (continuous, one shot), The timer clock prescaler, The timer period , The timer repetition counter.</li> </ul> |

### HAL\_HRTIM\_DLLCalibrationStart

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_DLLCalibrationStart (HRTIM_HandleTypeDef * hhrtim, uint32_t CalibrationRate)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Starts the DLL calibration.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>CalibrationRate:</b> DLL calibration period This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_SINGLE_CALIBRATION: One shot DLL calibration</li> <li>- HRTIM_CALIBRATIONRATE_7300: Periodic DLL calibration. T=7.3 ms</li> <li>- HRTIM_CALIBRATIONRATE_910: Periodic DLL calibration. T=910 us</li> <li>- HRTIM_CALIBRATIONRATE_114: Periodic DLL calibration. T=114 us</li> <li>- HRTIM_CALIBRATIONRATE_14: Periodic DLL calibration. T=14 us</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Notes                | <ul style="list-style-type: none"> <li>• This function locks the HRTIM instance. HRTIM instance is unlocked within the HAL_HRTIM_PollForDLLCalibration function, just before exiting the function.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                          |

### HAL\_HRTIM\_DLLCalibrationStart\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_DLLCalibrationStart_IT (HRTIM_HandleTypeDef * hhrtim, uint32_t CalibrationRate)</b>                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Starts the DLL calibration.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>CalibrationRate:</b> DLL calibration period This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_SINGLE_CALIBRATION: One shot DLL calibration</li> <li>- HRTIM_CALIBRATIONRATE_7300: Periodic DLL calibration. T=7.3 ms</li> <li>- HRTIM_CALIBRATIONRATE_910: Periodic DLL calibration. T=910 us</li> <li>- HRTIM_CALIBRATIONRATE_114: Periodic DLL calibration. T=114 us</li> </ul> </li> </ul> |

- HRTIM\_CALIBRATIONRATE\_14: Periodic DLL calibration. T=14 us

Return values

- **HAL:** status

Notes

- This function locks the HRTIM instance. HRTIM instance is unlocked within the IRQ processing function when processing the DLL ready interrupt.
- If this function is called for periodic calibration, the DLLRDY interrupt is generated every time the calibration completes which will significantly increases the overall interrupt rate.

### **HAL\_HRTIM\_PollForDLLCalibration**

Function name      **HAL\_StatusTypeDef HAL\_HRTIM\_PollForDLLCalibration(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t Timeout)**

Function description      Polls the DLL calibration ready flag and returns when the flag is set (DLL calibration completed) or upon timeout expiration.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **Timeout:** Timeout duration in millisecond

Return values

- **HAL:** status

### **HAL\_HRTIM\_SimpleBaseStart**

Function name      **HAL\_StatusTypeDef HAL\_HRTIM\_SimpleBaseStart(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description      Starts the counter of a timer operating in simple time base mode.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index. This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_MASTER for master timer
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values

- **HAL:** status

### **HAL\_HRTIM\_SimpleBaseStop**

Function name      **HAL\_StatusTypeDef HAL\_HRTIM\_SimpleBaseStop(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description      Stops the counter of a timer operating in simple time base mode.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index. This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_MASTER for master timer
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C

- HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E
- Return values
- **HAL:** status

### **HAL\_HRTIM\_SimpleBaseStart\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleBaseStart_IT<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx)</b>                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Starts the counter of a timer operating in simple time base mode (Timer repetition interrupt is enabled).                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_MASTER for master timer</li> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

### **HAL\_HRTIM\_SimpleBaseStop\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleBaseStop_IT<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx)</b>                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Stops the counter of a timer operating in simple time base mode (Timer repetition interrupt is disabled).                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_MASTER for master timer</li> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

### **HAL\_HRTIM\_SimpleBaseStart\_DMA**

|                      |                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleBaseStart_DMA<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t SrcAddr, uint32_t DestAddr, uint32_t Length)</b>                                                                                                                       |
| Function description | Starts the counter of a timer operating in simple time base mode (Timer repetition DMA request is enabled).                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_MASTER for master timer</li> </ul> </li> </ul> |

- HRTIM\_TIMERINDEX\_TIMER\_A for timer A
- HRTIM\_TIMERINDEX\_TIMER\_B for timer B
- HRTIM\_TIMERINDEX\_TIMER\_C for timer C
- HRTIM\_TIMERINDEX\_TIMER\_D for timer D
- HRTIM\_TIMERINDEX\_TIMER\_E for timer E
- **SrcAddr:** DMA transfer source address
- **DestAddr:** DMA transfer destination address
- **Length:** The length of data items (data size) to be transferred from source to destination

### **HAL\_HRTIM\_SimpleBaseStop\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleBaseStop_DMA(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Stops the counter of a timer operating in simple time base mode (Timer repetition DMA request is disabled).                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_MASTER for master timer</li> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

### **HAL\_HRTIM\_SimpleOCChannelConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleOCChannelConfig(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t OCChannel, HRTIM_SimpleOCChannelCfgTypeDef * pSimpleOCChannelCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Configures an output in simple output compare mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>OCChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> </ul> </li> </ul> |

|                                                                                                   |                                                                                                                                                                                          |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                                   | <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>- HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>- HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> |
| • <b>pSimpleOCChannelCfg:</b> pointer to the simple output compare output configuration structure |                                                                                                                                                                                          |
| Return values                                                                                     | • <b>HAL:</b> status                                                                                                                                                                     |

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes | <ul style="list-style-type: none"> <li>• When the timer operates in simple output compare mode:<br/>Output 1 is implicitly controlled by the compare unit 1<br/>Output 2 is implicitly controlled by the compare unit 2<br/>Output Set/Reset crossbar is set according to the selected output compare mode:<br/>Toggle: SETxyR = RSTxyR = CMPy Active:<br/>SETxyR = CMPy, RSTxyR = 0 Inactive: SETxy =0, RSTxy = CMPy</li> </ul> |
|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### HAL\_HRTIM\_SimpleOCStart

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleOCStart</b><br><b>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t OCChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Starts the output compare signal generation on the designed timer output.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>OCChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>- HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>- HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>- HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

### HAL\_HRTIM\_SimpleOCStop

|                      |                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleOCStop</b><br><b>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t OCChannel)</b> |
| Function description | Stops the output compare signal generation on the designed timer output.                                                        |

- |               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>OCChannel:</b> Timer output This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>- HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>- HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>- HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

### **HAL\_HRTIM\_SimpleOCStart\_IT**

- |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleOCStart_IT(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t OCChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Starts the output compare signal generation on the designed timer output (Interrupt is enabled (see note note below)).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>OCChannel:</b> Timer output This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>- HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>- HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>- HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

- |       |                                                                                                                                                                                                                                                                              |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes | <ul style="list-style-type: none"> <li>• Interrupt enabling depends on the chosen output compare mode Output toggle: compare match interrupt is enabled Output set active: output set interrupt is enabled Output set inactive: output reset interrupt is enabled</li> </ul> |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### **HAL\_HRTIM\_SimpleOCStop\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleOCStop_IT<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t OCChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Stops the output compare signal generation on the designed timer output (Interrupt is disabled).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>OCChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>- HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>- HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>- HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

### **HAL\_HRTIM\_SimpleOCStart\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleOCStart_DMA<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t OCChannel, uint32_t SrcAddr, uint32_t DestAddr, uint32_t Length)</b>                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Starts the output compare signal generation on the designed timer output (DMA request is enabled (see note below)).                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>OCChannel:</b> Timer output This parameter can be one of the</li> </ul> |

following values:

- HRTIM\_OUTPUT\_TA1: Timer A - Output 1
- HRTIM\_OUTPUT\_TA2: Timer A - Output 2
- HRTIM\_OUTPUT\_TB1: Timer B - Output 1
- HRTIM\_OUTPUT\_TB2: Timer B - Output 2
- HRTIM\_OUTPUT\_TC1: Timer C - Output 1
- HRTIM\_OUTPUT\_TC2: Timer C - Output 2
- HRTIM\_OUTPUT\_TD1: Timer D - Output 1
- HRTIM\_OUTPUT\_TD2: Timer D - Output 2
- HRTIM\_OUTPUT\_TE1: Timer E - Output 1
- HRTIM\_OUTPUT\_TE2: Timer E - Output 2
- **SrcAddr:** DMA transfer source address
- **DestAddr:** DMA transfer destination address
- **Length:** The length of data items (data size) to be transferred from source to destination

#### Return values

- **HAL:** status

#### Notes

- DMA request enabling depends on the chosen output compare mode Output toggle: compare match DMA request is enabled Output set active: output set DMA request is enabled Output set inactive: output reset DMA request is enabled

## HAL\_HRTIM\_SimpleOCStop\_DMA

#### Function name

**HAL\_StatusTypeDef HAL\_HRTIM\_SimpleOCStop\_DMA(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx, uint32\_t OCChannel)**

#### Function description

Stops the output compare signal generation on the designed timer output (DMA request is disabled).

#### Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E
- **OCChannel:** Timer output This parameter can be one of the following values:
  - HRTIM\_OUTPUT\_TA1: Timer A - Output 1
  - HRTIM\_OUTPUT\_TA2: Timer A - Output 2
  - HRTIM\_OUTPUT\_TB1: Timer B - Output 1
  - HRTIM\_OUTPUT\_TB2: Timer B - Output 2
  - HRTIM\_OUTPUT\_TC1: Timer C - Output 1
  - HRTIM\_OUTPUT\_TC2: Timer C - Output 2
  - HRTIM\_OUTPUT\_TD1: Timer D - Output 1
  - HRTIM\_OUTPUT\_TD2: Timer D - Output 2
  - HRTIM\_OUTPUT\_TE1: Timer E - Output 1
  - HRTIM\_OUTPUT\_TE2: Timer E - Output 2

#### Return values

- **HAL:** status

**HAL\_HRTIM\_SimplePWMChannelConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimplePWMChannelConfig<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>PWMChannel, HRTIM_SimplePWMChannelCfgTypeDef *<br/>pSimplePWMChannelCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Configures an output in simple PWM mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>PWMChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>– HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>– HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>– HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>– HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>– HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>– HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>– HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>– HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>– HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> <li>• <b>pSimplePWMChannelCfg:</b> pointer to the simple PWM output configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• When the timer operates in simple PWM output mode: Output 1 is implicitly controlled by the compare unit 1 Output 2 is implicitly controlled by the compare unit 2 Output Set/Reset crossbar is set as follows: Output 1: SETx1R = CMP1, RSTx1R = PER Output 2: SETx2R = CMP2, RST2R = PER</li> <li>• When Simple PWM mode is used the registers preload mechanism is enabled (otherwise the behavior is not guaranteed).</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

**HAL\_HRTIM\_SimplePWMStart**

|                      |                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimplePWMStart<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>PWMChannel)</b>                                                                                                                                                                                                    |
| Function description | Starts the PWM output signal generation on the designed timer output.                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> </ul> </li> </ul> |

- HRTIM\_TIMERINDEX\_TIMER\_C for timer C
- HRTIM\_TIMERINDEX\_TIMER\_D for timer D
- HRTIM\_TIMERINDEX\_TIMER\_E for timer E
- **PWMChannel:** Timer output This parameter can be one of the following values:
  - HRTIM\_OUTPUT\_TA1: Timer A - Output 1
  - HRTIM\_OUTPUT\_TA2: Timer A - Output 2
  - HRTIM\_OUTPUT\_TB1: Timer B - Output 1
  - HRTIM\_OUTPUT\_TB2: Timer B - Output 2
  - HRTIM\_OUTPUT\_TC1: Timer C - Output 1
  - HRTIM\_OUTPUT\_TC2: Timer C - Output 2
  - HRTIM\_OUTPUT\_TD1: Timer D - Output 1
  - HRTIM\_OUTPUT\_TD2: Timer D - Output 2
  - HRTIM\_OUTPUT\_TE1: Timer E - Output 1
  - HRTIM\_OUTPUT\_TE2: Timer E - Output 2

Return values

- **HAL:** status

### HAL\_HRTIM\_SimplePWMStop

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_HRTIM_SimplePWMStop<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>PWMChannel)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Stops the PWM output signal generation on the designed timer output.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>PWMChannel:</b> Timer output This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>- HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>- HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>- HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**HAL\_HRTIM\_SimplePWMStart\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimplePWMStart_IT<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>PWMChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description | Starts the PWM output signal generation on the designed timer output (The compare interrupt is enabled).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>PWMChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>– HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>– HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>– HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>– HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>– HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>– HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>– HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>– HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>– HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

**HAL\_HRTIM\_SimplePWMStop\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimplePWMStop_IT<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>PWMChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Stops the PWM output signal generation on the designed timer output (The compare interrupt is disabled).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>PWMChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>– HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>– HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>– HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>– HRTIM_OUTPUT_TC1: Timer C - Output 1</li> </ul> </li> </ul> |

- HRTIM\_OUTPUT\_TC2: Timer C - Output 2
- HRTIM\_OUTPUT\_TD1: Timer D - Output 1
- HRTIM\_OUTPUT\_TD2: Timer D - Output 2
- HRTIM\_OUTPUT\_TE1: Timer E - Output 1
- HRTIM\_OUTPUT\_TE2: Timer E - Output 2

Return values

- **HAL:** status

### **HAL\_HRTIM\_SimplePWMStart\_DMA**

Function name

**HAL\_StatusTypeDef HAL\_HRTIM\_SimplePWMStart\_DMA**  
**(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx, uint32\_t**  
**PWMChannel, uint32\_t SrcAddr, uint32\_t DestAddr, uint32\_t**  
**Length)**

Function description

Starts the PWM output signal generation on the designed timer output (The compare DMA request is enabled).

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E
- **PWMChannel:** Timer output This parameter can be one of the following values:
  - HRTIM\_OUTPUT\_TA1: Timer A - Output 1
  - HRTIM\_OUTPUT\_TA2: Timer A - Output 2
  - HRTIM\_OUTPUT\_TB1: Timer B - Output 1
  - HRTIM\_OUTPUT\_TB2: Timer B - Output 2
  - HRTIM\_OUTPUT\_TC1: Timer C - Output 1
  - HRTIM\_OUTPUT\_TC2: Timer C - Output 2
  - HRTIM\_OUTPUT\_TD1: Timer D - Output 1
  - HRTIM\_OUTPUT\_TD2: Timer D - Output 2
  - HRTIM\_OUTPUT\_TE1: Timer E - Output 1
  - HRTIM\_OUTPUT\_TE2: Timer E - Output 2
- **SrcAddr:** DMA transfer source address
- **DestAddr:** DMA transfer destination address
- **Length:** The length of data items (data size) to be transferred from source to destination

Return values

- **HAL:** status

### **HAL\_HRTIM\_SimplePWMStop\_DMA**

Function name

**HAL\_StatusTypeDef HAL\_HRTIM\_SimplePWMStop\_DMA**  
**(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx, uint32\_t**  
**PWMChannel)**

Function description

Stops the PWM output signal generation on the designed timer output (The compare DMA request is disabled).

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the

following values:

- HRTIM\_TIMERINDEX\_TIMER\_A for timer A
- HRTIM\_TIMERINDEX\_TIMER\_B for timer B
- HRTIM\_TIMERINDEX\_TIMER\_C for timer C
- HRTIM\_TIMERINDEX\_TIMER\_D for timer D
- HRTIM\_TIMERINDEX\_TIMER\_E for timer E
- **PWMChannel:** Timer output This parameter can be one of the following values:
  - HRTIM\_OUTPUT\_TA1: Timer A - Output 1
  - HRTIM\_OUTPUT\_TA2: Timer A - Output 2
  - HRTIM\_OUTPUT\_TB1: Timer B - Output 1
  - HRTIM\_OUTPUT\_TB2: Timer B - Output 2
  - HRTIM\_OUTPUT\_TC1: Timer C - Output 1
  - HRTIM\_OUTPUT\_TC2: Timer C - Output 2
  - HRTIM\_OUTPUT\_TD1: Timer D - Output 1
  - HRTIM\_OUTPUT\_TD2: Timer D - Output 2
  - HRTIM\_OUTPUT\_TE1: Timer E - Output 1
  - HRTIM\_OUTPUT\_TE2: Timer E - Output 2

#### Return values

- **HAL:** status

### **HAL\_HRTIM\_SimpleCaptureChannelConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef</b><br><b>HAL_HRTIM_SimpleCaptureChannelConfig</b><br><i>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t CaptureChannel, HRTIM_SimpleCaptureChannelCfgTypeDef * pSimpleCaptureChannelCfg)</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Configures a simple capture.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>CaptureChannel:</b> Capture unit This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_CAPTUREUNIT_1: Capture unit 1</li> <li>- HRTIM_CAPTUREUNIT_2: Capture unit 2</li> </ul> </li> <li>• <b>pSimpleCaptureChannelCfg:</b> pointer to the simple capture configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Notes                | <ul style="list-style-type: none"> <li>• When the timer operates in simple capture mode the capture is triggered by the designated external event and GPIO input is implicitly used as event source. The capture can be triggered by a rising edge, a falling edge or both edges on event channel.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

**HAL\_HRTIM\_SimpleCaptureStart**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleCaptureStart<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>CaptureChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description | Enables a simple capture on the designed capture unit.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>CaptureChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_CAPTUREUNIT_1: Capture unit 1</li> <li>– HRTIM_CAPTUREUNIT_2: Capture unit 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• The external event triggering the capture is available for all timing units. It can be used directly and is active as soon as the timing unit counter is enabled.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

**HAL\_HRTIM\_SimpleCaptureStop**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleCaptureStop<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>CaptureChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Disables a simple capture on the designed capture unit.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>CaptureChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_CAPTUREUNIT_1: Capture unit 1</li> <li>– HRTIM_CAPTUREUNIT_2: Capture unit 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

**HAL\_HRTIM\_SimpleCaptureStart\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleCaptureStart_IT<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t CaptureChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Enables a simple capture on the designed capture unit (Capture interrupt is enabled).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>CaptureChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_CAPTUREUNIT_1: Capture unit 1</li> <li>– HRTIM_CAPTUREUNIT_2: Capture unit 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

**HAL\_HRTIM\_SimpleCaptureStop\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleCaptureStop_IT<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t CaptureChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Disables a simple capture on the designed capture unit (Capture interrupt is disabled).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>CaptureChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_CAPTUREUNIT_1: Capture unit 1</li> <li>– HRTIM_CAPTUREUNIT_2: Capture unit 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

**HAL\_HRTIM\_SimpleCaptureStart\_DMA**

|                      |                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleCaptureStart_DMA<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t CaptureChannel, uint32_t SrcAddr, uint32_t DestAddr, uint32_t Length)</b> |
| Function description | Enables a simple capture on the designed capture unit (Capture DMA request is enabled).                                                                                                        |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>CaptureChannel:</b> Timer output This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_CAPTUREUNIT_1: Capture unit 1</li> <li>- HRTIM_CAPTUREUNIT_2: Capture unit 2</li> </ul> </li> <li>• <b>SrcAddr:</b> DMA transfer source address</li> <li>• <b>DestAddr:</b> DMA transfer destination address</li> <li>• <b>Length:</b> The length of data items (data size) to be transferred from source to destination</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

### HAL\_HRTIM\_SimpleCaptureStop\_DMA

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_HRTIM_SimpleCaptureStop_DMA(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t CaptureChannel)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Disables a simple capture on the designed capture unit (Capture DMA request is disabled).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>CaptureChannel:</b> Timer output This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_CAPTUREUNIT_1: Capture unit 1</li> <li>- HRTIM_CAPTUREUNIT_2: Capture unit 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**HAL\_HRTIM\_SimpleOnePulseChannelConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef<br/>HAL_HRTIM_SimpleOnePulseChannelConfig<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>OnePulseChannel,<br/>HRTIM_SimpleOnePulseChannelCfgTypeDef *<br/>pSimpleOnePulseChannelCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Configures an output simple one pulse mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>OnePulseChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>- HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>- HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>- HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> <li>• <b>pSimpleOnePulseChannelCfg:</b> pointer to the simple one pulse output configuration structure</li> </ul> |
| Return values        | <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• When the timer operates in simple one pulse mode: the timer counter is implicitly started by the reset event, the reset of the timer counter is triggered by the designated external event GPIO input is implicitly used as event source, Output 1 is implicitly controlled by the compare unit 1, Output 2 is implicitly controlled by the compare unit 2. Output Set/Reset crossbar is set as follows: Output 1: SETx1R = CMP1, RSTx1R = PER Output 2: SETx2R = CMP2, RST2R = PER</li> <li>• If HAL_HRTIM_SimpleOnePulseChannelConfig is called for both timer outputs, the reset event related configuration data provided in the second call will override the reset event related configuration data provided in the first call.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

**HAL\_HRTIM\_SimpleOnePulseStart**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleOnePulseStart<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>OnePulseChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Enables the simple one pulse signal generation on the designed output.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>OnePulseChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>– HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>– HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>– HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>– HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>– HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>– HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>– HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>– HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>– HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

**HAL\_HRTIM\_SimpleOnePulseStop**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SimpleOnePulseStop<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>OnePulseChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Disables the simple one pulse signal generation on the designed output.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>OnePulseChannel:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>– HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>– HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>– HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>– HRTIM_OUTPUT_TC1: Timer C - Output 1</li> </ul> </li> </ul> |

- HRTIM\_OUTPUT\_TC2: Timer C - Output 2
- HRTIM\_OUTPUT\_TD1: Timer D - Output 1
- HRTIM\_OUTPUT\_TD2: Timer D - Output 2
- HRTIM\_OUTPUT\_TE1: Timer E - Output 1
- HRTIM\_OUTPUT\_TE2: Timer E - Output 2

Return values

- **HAL:** status

**HAL\_HRTIM\_SimpleOnePulseStart\_IT**

Function name

**HAL\_StatusTypeDef HAL\_HRTIM\_SimpleOnePulseStart\_IT  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx, uint32\_t OnePulseChannel)**

Function description

Enables the simple one pulse signal generation on the designed output (The compare interrupt is enabled (pulse start)).

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E
- **OnePulseChannel:** Timer output This parameter can be one of the following values:
  - HRTIM\_OUTPUT\_TA1: Timer A - Output 1
  - HRTIM\_OUTPUT\_TA2: Timer A - Output 2
  - HRTIM\_OUTPUT\_TB1: Timer B - Output 1
  - HRTIM\_OUTPUT\_TB2: Timer B - Output 2
  - HRTIM\_OUTPUT\_TC1: Timer C - Output 1
  - HRTIM\_OUTPUT\_TC2: Timer C - Output 2
  - HRTIM\_OUTPUT\_TD1: Timer D - Output 1
  - HRTIM\_OUTPUT\_TD2: Timer D - Output 2
  - HRTIM\_OUTPUT\_TE1: Timer E - Output 1
  - HRTIM\_OUTPUT\_TE2: Timer E - Output 2

Return values

- **HAL:** status

**HAL\_HRTIM\_SimpleOnePulseStop\_IT**

Function name

**HAL\_StatusTypeDef HAL\_HRTIM\_SimpleOnePulseStop\_IT  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx, uint32\_t OnePulseChannel)**

Function description

Disables the simple one pulse signal generation on the designed output (The compare interrupt is disabled).

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D

- HRTIM\_TIMERINDEX\_TIMER\_E for timer E
- **OnePulseChannel:** Timer output This parameter can be one of the following values:
  - HRTIM\_OUTPUT\_TA1: Timer A - Output 1
  - HRTIM\_OUTPUT\_TA2: Timer A - Output 2
  - HRTIM\_OUTPUT\_TB1: Timer B - Output 1
  - HRTIM\_OUTPUT\_TB2: Timer B - Output 2
  - HRTIM\_OUTPUT\_TC1: Timer C - Output 1
  - HRTIM\_OUTPUT\_TC2: Timer C - Output 2
  - HRTIM\_OUTPUT\_TD1: Timer D - Output 1
  - HRTIM\_OUTPUT\_TD2: Timer D - Output 2
  - HRTIM\_OUTPUT\_TE1: Timer E - Output 1
  - HRTIM\_OUTPUT\_TE2: Timer E - Output 2

**Return values**

- **HAL:** status

**HAL\_HRTIM\_BurstModeConfig**

|                      |                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_BurstModeConfig(HRTIM_HandleTypeDef * hhrtim, HRTIM_BurstModeCfgTypeDef * pBurstModeCfg)</b>                                                       |
| Function description | Configures the burst mode feature of the HRTIM.                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>pBurstModeCfg:</b> pointer to the burst mode configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                            |
| Notes                | <ul style="list-style-type: none"> <li>• This function must be called before starting the burst mode controller</li> </ul>                                                        |

**HAL\_HRTIM\_EventConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_EventConfig(HRTIM_HandleTypeDef * hhrtim, uint32_t Event, HRTIM_EventCfgTypeDef * pEventCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Configures the conditioning of an external event.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>Event:</b> external event to configure This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_EVENT_1: External event 1</li> <li>- HRTIM_EVENT_2: External event 2</li> <li>- HRTIM_EVENT_3: External event 3</li> <li>- HRTIM_EVENT_4: External event 4</li> <li>- HRTIM_EVENT_5: External event 5</li> <li>- HRTIM_EVENT_6: External event 6</li> <li>- HRTIM_EVENT_7: External event 7</li> <li>- HRTIM_EVENT_8: External event 8</li> <li>- HRTIM_EVENT_9: External event 9</li> <li>- HRTIM_EVENT_10: External event 10</li> </ul> </li> <li>• <b>pEventCfg:</b> pointer to the event conditioning configuration structure</li> </ul> |

---

|               |                                                                                                          |
|---------------|----------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                     |
| Notes         | <ul style="list-style-type: none"> <li>This function must be called before starting the timer</li> </ul> |

### HAL\_HRTIM\_EventPrescalerConfig

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_EventPrescalerConfig<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t Prescaler)</b>                                                                                                                                                                                                                                                                                                                                       |
| Function description | Configures the external event conditioning block prescaler.                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li><b>Prescaler:</b> Prescaler value This parameter can be one of the following values: <ul style="list-style-type: none"> <li>HRTIM_EVENTPRESCALER_DIV1: fEEVS=fHRTIM</li> <li>HRTIM_EVENTPRESCALER_DIV2: fEEVS=fHRTIM / 2</li> <li>HRTIM_EVENTPRESCALER_DIV4: fEEVS=fHRTIM / 4</li> <li>HRTIM_EVENTPRESCALER_DIV8: fEEVS=fHRTIM / 8</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                 |
| Notes                | <ul style="list-style-type: none"> <li>This function must be called before starting the timer</li> </ul>                                                                                                                                                                                                                                                                                                                                             |

### HAL\_HRTIM\_FaultConfig

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_FaultConfig<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t Fault,<br/>HRTIM_FaultCfgTypeDef * pFaultCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Configures the conditioning of fault input.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li><b>Fault:</b> fault input to configure This parameter can be one of the following values: <ul style="list-style-type: none"> <li>HRTIM_FAULT_1: Fault input 1</li> <li>HRTIM_FAULT_2: Fault input 2</li> <li>HRTIM_FAULT_3: Fault input 3</li> <li>HRTIM_FAULT_4: Fault input 4</li> <li>HRTIM_FAULT_5: Fault input 5</li> </ul> </li> <li><b>pFaultCfg:</b> pointer to the fault conditioning configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Notes                | <ul style="list-style-type: none"> <li>This function must be called before starting the timer and before enabling faults inputs</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                   |

### HAL\_HRTIM\_FaultPrescalerConfig

|                      |                                                                                                                                                                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_FaultPrescalerConfig<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t Prescaler)</b>                                                                                                                                                                                                                             |
| Function description | Configures the fault conditioning block prescaler.                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li><b>Prescaler:</b> Prescaler value This parameter can be one of the following values: <ul style="list-style-type: none"> <li>HRTIM_FAULTPRESCALER_DIV1: fFLTS=fHRTIM</li> <li>HRTIM_FAULTPRESCALER_DIV2: fFLTS=fHRTIM / 2</li> </ul> </li> </ul> |

|               |                                                                                                                                                        |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>– HRTIM_FAULTPRESCALER_DIV4: fFLTS=fHRTIM / 4</li> <li>– HRTIM_FAULTPRESCALER_DIV8: fFLTS=fHRTIM / 8</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                 |
| Notes         | <ul style="list-style-type: none"> <li>• This function must be called before starting the timer and before enabling faults inputs</li> </ul>           |

### HAL\_HRTIM\_FaultModeCtl

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void HAL_HRTIM_FaultModeCtl (HRTIM_HandleTypeDef * hhrtim, uint32_t Faults, uint32_t Enable)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Enables or disables the HRTIMx Fault mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>Faults:</b> fault input(s) to enable or disable This parameter can be any combination of the following values: <ul style="list-style-type: none"> <li>– HRTIM_FAULT_1: Fault input 1</li> <li>– HRTIM_FAULT_2: Fault input 2</li> <li>– HRTIM_FAULT_3: Fault input 3</li> <li>– HRTIM_FAULT_4: Fault input 4</li> <li>– HRTIM_FAULT_5: Fault input 5</li> </ul> </li> <li>• <b>Enable:</b> Fault(s) enabling This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_FAULTMODECTL_ENABLED: Fault(s) enabled</li> <li>– HRTIM_FAULTMODECTL_DISABLED: Fault(s) disabled</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

### HAL\_HRTIM\_ADCTriggerConfig

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_HRTIM_ADCTriggerConfig (HRTIM_HandleTypeDef * hhrtim, uint32_t ADCTrigger, HRTIM_ADCTriggerCfgTypeDef * pADCTriggerCfg)</code>                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Configures both the ADC trigger register update source and the ADC trigger source.                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>ADCTrigger:</b> ADC trigger to configure This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_ADCTRIGGER_1: ADC trigger 1</li> <li>– HRTIM_ADCTRIGGER_2: ADC trigger 2</li> <li>– HRTIM_ADCTRIGGER_3: ADC trigger 3</li> <li>– HRTIM_ADCTRIGGER_4: ADC trigger 4</li> </ul> </li> <li>• <b>pADCTriggerCfg:</b> pointer to the ADC trigger configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• This function must be called before starting the timer</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                  |

### **HAL\_HRTIM\_WaveformTimerConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_WaveformTimerConfig(<br/>HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx,<br/>HRTIM_TimerCfgTypeDef * pTimerCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Configures the general behavior of a timer operating in waveform mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_MASTER for master timer</li> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>pTimerCfg:</b> pointer to the timer configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Notes                | <ul style="list-style-type: none"> <li>• When the timer operates in waveform mode, all the features supported by the HRTIM are available without any limitation.</li> <li>• This function must be called before starting the timer</li> </ul>                                                                                                                                                                                                                                                                                                                                                                     |

### **HAL\_HRTIM\_WaveformCompareConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_WaveformCompareConfig(<br/>HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>CompareUnit, HRTIM_CompareCfgTypeDef * pCompareCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Configures the compare unit of a timer operating in waveform mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_MASTER for master timer</li> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>CompareUnit:</b> Compare unit to configure This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_COMPAREUNIT_1: Compare unit 1</li> <li>– HRTIM_COMPAREUNIT_2: Compare unit 2</li> <li>– HRTIM_COMPAREUNIT_3: Compare unit 3</li> <li>– HRTIM_COMPAREUNIT_4: Compare unit 4</li> </ul> </li> <li>• <b>pCompareCfg:</b> pointer to the compare unit configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• When auto delayed mode is required for compare unit 2 or compare unit 4, application has to configure separately the capture unit. Capture unit to configure in that case depends</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

on the compare unit auto delayed mode is applied to (see below): Auto delayed on output compare 2: capture unit 1 must be configured Auto delayed on output compare 4: capture unit 2 must be configured

- This function must be called before starting the timer

### **HAL\_HRTIM\_WaveformCaptureConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_WaveformCaptureConfig(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t CaptureUnit, HRTIM_CaptureCfgTypeDef * pCaptureCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Configures the capture unit of a timer operating in waveform mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>CaptureUnit:</b> Capture unit to configure This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_CAPTUREUNIT_1: Capture unit 1</li> <li>– HRTIM_CAPTUREUNIT_2: Capture unit 2</li> </ul> </li> <li>• <b>pCaptureCfg:</b> pointer to the compare unit configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This function must be called before starting the timer</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

### **HAL\_HRTIM\_WaveformOutputConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_WaveformOutputConfig(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t Output, HRTIM_OutputCfgTypeDef * pOutputCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Configures the output of a timer operating in waveform mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>Output:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>– HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>– HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>– HRTIM_OUTPUT_TB2: Timer B - Output 2</li> </ul> </li> </ul> |

- HRTIM\_OUTPUT\_TC1: Timer C - Output 1
  - HRTIM\_OUTPUT\_TC2: Timer C - Output 2
  - HRTIM\_OUTPUT\_TD1: Timer D - Output 1
  - HRTIM\_OUTPUT\_TD2: Timer D - Output 2
  - HRTIM\_OUTPUT\_TE1: Timer E - Output 1
  - HRTIM\_OUTPUT\_TE2: Timer E - Output 2
  - **pOutputCfg:** pointer to the timer output configuration structure
- Return values**
- **HAL:** status
- Notes**
- This function must be called before configuring the timer and after configuring the deadtime insertion feature (if required).

### HAL\_HRTIM\_WaveformSetOutputLevel

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_WaveformSetOutputLevel(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t Output, uint32_t OutputLevel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Forces the timer output to its active or inactive state.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>Output:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>- HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>- HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>- HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> <li>• <b>OutputLevel:</b> indicates whether the output is forced to its active or inactive level This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_OUTPUTLEVEL_ACTIVE: output is forced to its active level</li> <li>- HRTIM_OUTPUTLEVEL_INACTIVE: output is forced to its inactive level</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• The 'software set/reset trigger' bit in the output set/reset registers is automatically reset by hardware</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

**HAL\_HRTIM\_TimerEventFilteringConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_TimerEventFilteringConfig<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>Event, HRTIM_TimerEventFilteringCfgTypeDef *<br/>pTimerEventFilteringCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Configures the event filtering capabilities of a timer (blanking, windowing)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>Event:</b> external event for which timer event filtering must be configured This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_EVENT_NONE Reset timer event filtering configuration</li> <li>– HRTIM_EVENT_1: External event 1</li> <li>– HRTIM_EVENT_2: External event 2</li> <li>– HRTIM_EVENT_3: External event 3</li> <li>– HRTIM_EVENT_4: External event 4</li> <li>– HRTIM_EVENT_5: External event 5</li> <li>– HRTIM_EVENT_6: External event 6</li> <li>– HRTIM_EVENT_7: External event 7</li> <li>– HRTIM_EVENT_8: External event 8</li> <li>– HRTIM_EVENT_9: External event 9</li> <li>– HRTIM_EVENT_10: External event 10</li> </ul> </li> <li>• <b>pTimerEventFilteringCfg:</b> pointer to the timer event filtering configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• This function must be called before starting the timer</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

**HAL\_HRTIM\_DeadTimeConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_DeadTimeConfig<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx,<br/>HRTIM_DeadTimeCfgTypeDef * pDeadTimeCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Configures the deadtime insertion feature for a timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>pDeadTimeCfg:</b> pointer to the deadtime insertion</li> </ul> |

configuration structure

- |               |                                                                                                            |
|---------------|------------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                     |
| Notes         | <ul style="list-style-type: none"> <li>• This function must be called before starting the timer</li> </ul> |

### **HAL\_HRTIM\_ChopperModeConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_ChopperModeConfig<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx,<br/>HRTIM_ChopperModeCfgTypeDef * pChopperModeCfg)</b>                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Configures the chopper mode feature for a timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>pChopperModeCfg:</b> pointer to the chopper mode configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This function must be called before configuring the timer output(s)</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

### **HAL\_HRTIM\_BurstDMAConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_BurstDMAConfig<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t<br/>RegistersToUpdate)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Configures the burst DMA controller for a timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_MASTER for master timer</li> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>RegistersToUpdate:</b> registers to be written by DMA This parameter can be any combination of the following values: <ul style="list-style-type: none"> <li>- HRTIM_BURSTDMA_CR: HRTIM_MCR or HRTIM_TIMxCR</li> <li>- HRTIM_BURSTDMA_ICR: HRTIM_MICR or HRTIM_TIMxICR</li> <li>- HRTIM_BURSTDMA_DIER: HRTIM_MDIER or HRTIM_TIMxDIER</li> <li>- HRTIM_BURSTDMA_CNT: HRTIM_MCNT or HRTIM_TIMxCNT</li> </ul> </li> </ul> |

- HRTIM\_BURSTDMA\_PER: HRTIM\_MP PER or HRTIM\_TIMxPER
- HRTIM\_BURSTDMA REP: HRTIM\_MP REP or HRTIM\_TIMxREP
- HRTIM\_BURSTDMA\_CMP1: HRTIM\_MP CMP1 or HRTIM\_TIMxCMP1
- HRTIM\_BURSTDMA\_CMP2: HRTIM\_MP CMP2 or HRTIM\_TIMxCMP2
- HRTIM\_BURSTDMA\_CMP3: HRTIM\_MP CMP3 or HRTIM\_TIMxCMP3
- HRTIM\_BURSTDMA\_CMP4: HRTIM\_MP CMP4 or HRTIM\_TIMxCMP4
- HRTIM\_BURSTDMA\_DTR: HRTIM\_TIMxDTR
- HRTIM\_BURSTDMA\_SET1R: HRTIM\_TIMxSET1R
- HRTIM\_BURSTDMA\_RST1R: HRTIM\_TIMxRST1R
- HRTIM\_BURSTDMA\_SET2R: HRTIM\_TIMxSET2R
- HRTIM\_BURSTDMA\_RST2R: HRTIM\_TIMxRST2R
- HRTIM\_BURSTDMA\_EEFR1: HRTIM\_TIMxEEFR1
- HRTIM\_BURSTDMA\_EEFR2: HRTIM\_TIMxEEFR2
- HRTIM\_BURSTDMA\_RSTR: HRTIM\_TIMxRSTR
- HRTIM\_BURSTDMA\_CHPR: HRTIM\_TIMxCHPR
- HRTIM\_BURSTDMA\_OUTR: HRTIM\_TIMxOUTR
- HRTIM\_BURSTDMA\_FLTR: HRTIM\_TIMxFLTR

**Return values**

- **HAL:** status

**Notes**

- This function must be called before starting the timer

### HAL\_HRTIM\_WaveformCounterStart

**Function name** **HAL\_StatusTypeDef HAL\_HRTIM\_WaveformCounterStart (HRTIM\_HandleTypeDef \* hhrtim, uint32\_t Timers)**

**Function description** Starts the counter of the designated timer(s) operating in waveform mode. Timers can be combined (ORed) to allow for simultaneous counter start.

**Parameters**

- **hhrtim:** pointer to HAL HRTIM handle
- **Timers:** Timer counter(s) to start. This parameter can be any combination of the following values:
  - HRTIM\_TIMERID\_MASTER
  - HRTIM\_TIMERID\_TIMER\_A
  - HRTIM\_TIMERID\_TIMER\_B
  - HRTIM\_TIMERID\_TIMER\_C
  - HRTIM\_TIMERID\_TIMER\_D
  - HRTIM\_TIMERID\_TIMER\_E

**Return values**

- **HAL:** status

**HAL\_HRTIM\_WaveformCounterStop**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_WaveformCounterStop<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t Timers)</b>                                                                                                                                                                                                                                                                                                                        |
| Function description | Stops the counter of the designated timer(s) operating in waveform mode Timers can be combined (ORed) to allow for simultaneous counter stop.                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>Timers:</b> Timer counter(s) to stop This parameter can be any combination of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERID_MASTER</li> <li>– HRTIM_TIMERID_A</li> <li>– HRTIM_TIMERID_B</li> <li>– HRTIM_TIMERID_C</li> <li>– HRTIM_TIMERID_D</li> <li>– HRTIM_TIMERID_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                            |
| Notes                | <ul style="list-style-type: none"> <li>• The counter of a timer is stopped only if all timer outputs are disabled</li> </ul>                                                                                                                                                                                                                                                                                                      |

**HAL\_HRTIM\_WaveformCounterStart\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_WaveformCounterStart_IT<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t Timers)</b>                                                                                                                                                                                                                                                                                                                     |
| Function description | Starts the counter of the designated timer(s) operating in waveform mode Timers can be combined (ORed) to allow for simultaneous counter start.                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>Timers:</b> Timer counter(s) to start This parameter can be any combination of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERID_MASTER</li> <li>– HRTIM_TIMERID_A</li> <li>– HRTIM_TIMERID_B</li> <li>– HRTIM_TIMERID_C</li> <li>– HRTIM_TIMERID_D</li> <li>– HRTIM_TIMERID_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• HRTIM interrupts (e.g. faults interrupts) and interrupts related to the timers to start are enabled within this function. Interrupts to enable are selected through HAL_HRTIM_WaveformTimerConfig function.</li> </ul>                                                                                                                                                                    |

**HAL\_HRTIM\_WaveformCounterStop\_IT**

|                      |                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_WaveformCounterStop_IT<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t Timers)</b>                                 |
| Function description | Stops the counter of the designated timer(s) operating in waveform mode Timers can be combined (ORed) to allow for simultaneous counter stop. |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li><b>Timers:</b> Timer counter(s) to stop This parameter can be any combination of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_TIMERID_MASTER</li> <li>- HRTIM_TIMERID_A</li> <li>- HRTIM_TIMERID_B</li> <li>- HRTIM_TIMERID_C</li> <li>- HRTIM_TIMERID_D</li> <li>- HRTIM_TIMERID_E</li> </ul> </li> </ul> |
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                    |
| Notes         | <ul style="list-style-type: none"> <li>The counter of a timer is stopped only if all timer outputs are disabled</li> <li>All enabled timer related interrupts are disabled.</li> </ul>                                                                                                                                                                                                                                                  |

### HAL\_HRTIM\_WaveformCounterStart\_DMA

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef</b><br><b>HAL_HRTIM_WaveformCounterStart_DMA</b><br><b>(HRTIM_HandleTypeDef * hhrtim, uint32_t Timers)</b>                                                                                                                                                                                                                                                                                                          |
| Function description | Starts the counter of the designated timer(s) operating in waveform mode Timers can be combined (ORed) to allow for simultaneous counter start.                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li><b>Timers:</b> Timer counter(s) to start This parameter can be any combination of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_TIMERID_MASTER</li> <li>- HRTIM_TIMERID_A</li> <li>- HRTIM_TIMERID_B</li> <li>- HRTIM_TIMERID_C</li> <li>- HRTIM_TIMERID_D</li> <li>- HRTIM_TIMERID_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>This function enables the dma request(s) mentionned in the timer configuration data structure for every timers to start.</li> <li>The source memory address, the destination memory address and the size of each DMA transfer are specified at timer configuration time (see <b>HAL_HRTIM_WaveformTimerConfig</b>)</li> </ul>                                                                     |

### HAL\_HRTIM\_WaveformCounterStop\_DMA

|                      |                                                                                                                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef</b><br><b>HAL_HRTIM_WaveformCounterStop_DMA</b><br><b>(HRTIM_HandleTypeDef * hhrtim, uint32_t Timers)</b>                                                                               |
| Function description | Stops the counter of the designated timer(s) operating in waveform mode Timers can be combined (ORed) to allow for simultaneous counter stop.                                                                |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li><b>Timers:</b> Timer counter(s) to stop This parameter can be any combination of the following values:</li> </ul> |

|               |                                                                                                                                                                                                                 |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>- HRTIM_TIMERID_MASTER</li> <li>- HRTIM_TIMERID_A</li> <li>- HRTIM_TIMERID_B</li> <li>- HRTIM_TIMERID_C</li> <li>- HRTIM_TIMERID_D</li> <li>- HRTIM_TIMERID_E</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                          |
| Notes         | <ul style="list-style-type: none"> <li>• The counter of a timer is stopped only if all timer outputs are disabled</li> <li>• All enabled timer related DMA requests are disabled.</li> </ul>                    |

### HAL\_HRTIM\_WaveformOutputStart

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_WaveformOutputStart (HRTIM_HandleTypeDef * hhrtim, uint32_t OutputsToStart)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description | Enables the generation of the waveform signal on the designated output(s) Outputs can be combined (ORed) to allow for simultaneous output enabling.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>OutputsToStart:</b> Timer output(s) to enable This parameter can be any combination of the following values: <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>- HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>- HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>- HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

### HAL\_HRTIM\_WaveformOutputStop

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_WaveformOutputStop (HRTIM_HandleTypeDef * hhrtim, uint32_t OutputsToStop)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Disables the generation of the waveform signal on the designated output(s) Outputs can be combined (ORed) to allow for simultaneous output disabling.                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>OutputsToStop:</b> Timer output(s) to disable This parameter can be any combination of the following values: <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> </ul> </li> </ul> |

- HRTIM\_OUTPUT\_TD2: Timer D - Output 2
- HRTIM\_OUTPUT\_TE1: Timer E - Output 1
- HRTIM\_OUTPUT\_TE2: Timer E - Output 2

Return values

- **HAL:** status

### **HAL\_HRTIM\_BurstModeCtl**

Function name

**HAL\_StatusTypeDef HAL\_HRTIM\_BurstModeCtl  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t Enable)**

Function description

Enables or disables the HRTIM burst mode controller.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **Enable:** Burst mode controller enabling This parameter can be one of the following values:
  - HRTIM\_BURSTMODECTL\_ENABLED: Burst mode enabled
  - HRTIM\_BURSTMODECTL\_DISABLED: Burst mode disabled

Return values

- **HAL:** status

Notes

- This function must be called after starting the timer(s)

### **HAL\_HRTIM\_BurstModeSoftwareTrigger**

Function name

**HAL\_StatusTypeDef HAL\_HRTIM\_BurstModeSoftwareTrigger  
(HRTIM\_HandleTypeDef \* hhrtim)**

Function description

Triggers the burst mode operation.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle

Return values

- **HAL:** status

### **HAL\_HRTIM\_SoftwareCapture**

Function name

**HAL\_StatusTypeDef HAL\_HRTIM\_SoftwareCapture  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx, uint32\_t CaptureUnit)**

Function description

Triggers a software capture on the designed capture unit.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E
- **CaptureUnit:** Capture unit to trig This parameter can be one of the following values:
  - HRTIM\_CAPTUREUNIT\_1: Capture unit 1
  - HRTIM\_CAPTUREUNIT\_2: Capture unit 2

Return values

- **HAL:** status

- Notes**
- The software capture' bit in the capure configuration register is automatically reset by hardware

### HAL\_HRTIM\_SoftwareUpdate

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SoftwareUpdate<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t Timers)</b>                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Triggers the update of the registers of one or several timers.                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li><b>Timers:</b> timers concerned with the software register update This parameter can be any combination of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERUPDATE_MASTER</li> <li>- HRTIM_TIMERUPDATE_A</li> <li>- HRTIM_TIMERUPDATE_B</li> <li>- HRTIM_TIMERUPDATE_C</li> <li>- HRTIM_TIMERUPDATE_D</li> <li>- HRTIM_TIMERUPDATE_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                            |
| Notes                | <ul style="list-style-type: none"> <li>The 'software update' bits in the HRTIM conrol register 2 register are automatically reset by hardware</li> </ul>                                                                                                                                                                                                                                                                                                                        |

### HAL\_HRTIM\_SoftwareReset

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_SoftwareReset<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t Timers)</b>                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Triggers the reset of one or several timers.                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li><b>Timers:</b> timers concerned with the software counter reset This parameter can be any combination of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERRESET_MASTER</li> <li>- HRTIM_TIMERRESET_TIMER_A</li> <li>- HRTIM_TIMERRESET_TIMER_B</li> <li>- HRTIM_TIMERRESET_TIMER_C</li> <li>- HRTIM_TIMERRESET_TIMER_D</li> <li>- HRTIM_TIMERRESET_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Notes                | <ul style="list-style-type: none"> <li>The 'software reset' bits in the HRTIM conrol register 2 are automatically reset by hardware</li> </ul>                                                                                                                                                                                                                                                                                                                                                        |

### HAL\_HRTIM\_BurstDMATransfer

|                      |                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_BurstDMATransfer<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t BurstBufferAddress, uint32_t BurstBufferLength)</b> |
| Function description | Starts a burst DMA operation to update HRTIM control registers content.                                                                                            |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_MASTER for master timer</li> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>BurstBufferAddress:</b> address of the buffer the HRTIM control registers content will be updated from.</li> <li>• <b>BurstBufferLength:</b> size (in WORDS) of the burst buffer.</li> </ul> |
| Return values | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes         | <ul style="list-style-type: none"> <li>• The TimerIdx parameter determines the dma channel to be used by the DMA burst controller (see below) <ul style="list-style-type: none"> <li>HRTIM_TIMERINDEX_MASTER: DMA channel 2 is used by the DMA burst controller</li> <li>HRTIM_TIMERINDEX_TIMER_A: DMA channel 3 is used by the DMA burst controller</li> <li>HRTIM_TIMERINDEX_TIMER_B: DMA channel 4 is used by the DMA burst controller</li> <li>HRTIM_TIMERINDEX_TIMER_C: DMA channel 5 is used by the DMA burst controller</li> <li>HRTIM_TIMERINDEX_TIMER_D: DMA channel 6 is used by the DMA burst controller</li> <li>HRTIM_TIMERINDEX_TIMER_E: DMA channel 7 is used by the DMA burst controller</li> </ul> </li> </ul>         |

### HAL\_HRTIM\_UpdateEnable

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_UpdateEnable<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t Timers)</b>                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Enables the transfer from preload to active registers for one or several timing units (including master timer).                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>Timers:</b> Timer(s) concerned by the register preload enabling command This parameter can be any combination of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERUPDATE_MASTER</li> <li>– HRTIM_TIMERUPDATE_A</li> <li>– HRTIM_TIMERUPDATE_B</li> <li>– HRTIM_TIMERUPDATE_C</li> <li>– HRTIM_TIMERUPDATE_D</li> <li>– HRTIM_TIMERUPDATE_E</li> </ul> </li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

### HAL\_HRTIM\_UpdateDisable

|                      |                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HRTIM_UpdateDisable<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t Timers)</b>             |
| Function description | Disables the transfer from preload to active registers for one or several timing units (including master timer). |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul>                   |

- **Timers:** Timer(s) concerned by the register preload disabling command This parameter can be any combination of the following values:
    - HRTIM\_TIMERUPDATE\_MASTER
    - HRTIM\_TIMERUPDATE\_A
    - HRTIM\_TIMERUPDATE\_B
    - HRTIM\_TIMERUPDATE\_C
    - HRTIM\_TIMERUPDATE\_D
    - HRTIM\_TIMERUPDATE\_E
- Return values
- **HAL:** status

### **HAL\_HRTIM\_GetState**

Function name      **HAL\_HRTIM\_StateTypeDef HAL\_HRTIM\_GetState  
(HRTIM\_HandleTypeDef \* hhrtim)**

Function description      return the HRTIM HAL state

Parameters      • **hhrtim:** pointer to HAL HRTIM handle

Return values      • **HAL:** state

### **HAL\_HRTIM\_GetCapturedValue**

Function name      **uint32\_t HAL\_HRTIM\_GetCapturedValue  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx, uint32\_t  
CaptureUnit)**

Function description      Returns actual value of the capture register of the designated capture unit.

Parameters      • **hhrtim:** pointer to HAL HRTIM handle  
                   • **TimerIdx:** Timer index This parameter can be one of the following values:  
                       – HRTIM\_TIMERINDEX\_TIMER\_A for timer A  
                       – HRTIM\_TIMERINDEX\_TIMER\_B for timer B  
                       – HRTIM\_TIMERINDEX\_TIMER\_C for timer C  
                       – HRTIM\_TIMERINDEX\_TIMER\_D for timer D  
                       – HRTIM\_TIMERINDEX\_TIMER\_E for timer E  
                   • **CaptureUnit:** Capture unit to trig This parameter can be one of the following values:  
                       – HRTIM\_CAPTUREUNIT\_1: Capture unit 1  
                       – HRTIM\_CAPTUREUNIT\_2: Capture unit 2

Return values      • **Captured:** value

### **HAL\_HRTIM\_WaveformGetOutputLevel**

Function name      **uint32\_t HAL\_HRTIM\_WaveformGetOutputLevel  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx, uint32\_t  
Output)**

Function description      Returns actual level (active or inactive) of the designated output.

Parameters      • **hhrtim:** pointer to HAL HRTIM handle  
                   • **TimerIdx:** Timer index This parameter can be one of the following values:

- HRTIM\_TIMERINDEX\_TIMER\_A for timer A
- HRTIM\_TIMERINDEX\_TIMER\_B for timer B
- HRTIM\_TIMERINDEX\_TIMER\_C for timer C
- HRTIM\_TIMERINDEX\_TIMER\_D for timer D
- HRTIM\_TIMERINDEX\_TIMER\_E for timer E
- **Output:** Timer output This parameter can be one of the following values:
  - HRTIM\_OUTPUT\_TA1: Timer A - Output 1
  - HRTIM\_OUTPUT\_TA2: Timer A - Output 2
  - HRTIM\_OUTPUT\_TB1: Timer B - Output 1
  - HRTIM\_OUTPUT\_TB2: Timer B - Output 2
  - HRTIM\_OUTPUT\_TC1: Timer C - Output 1
  - HRTIM\_OUTPUT\_TC2: Timer C - Output 2
  - HRTIM\_OUTPUT\_TD1: Timer D - Output 1
  - HRTIM\_OUTPUT\_TD2: Timer D - Output 2
  - HRTIM\_OUTPUT\_TE1: Timer E - Output 1
  - HRTIM\_OUTPUT\_TE2: Timer E - Output 2

Return values

- **Output:** level

Notes

- Returned output level is taken before the output stage (chopper, polarity).

### **HAL\_HRTIM\_WaveformGetOutputState**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_HRTIM_WaveformGetOutputState<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t Output)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Returns actual state (RUN, IDLE, FAULT) of the designated output.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>Output:</b> Timer output This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>- HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>- HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>- HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>- HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>- HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>- HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>- HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>- HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>- HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> </ul> |
| Return values        | • <b>Output:</b> state                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

**HAL\_HRTIM\_GetDelayedProtectionStatus**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_HRTIM_GetDelayedProtectionStatus<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx, uint32_t Output)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Returns the level (active or inactive) of the designated output when the delayed protection was triggered.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> <li>• <b>Output:</b> Timer output This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_OUTPUT_TA1: Timer A - Output 1</li> <li>– HRTIM_OUTPUT_TA2: Timer A - Output 2</li> <li>– HRTIM_OUTPUT_TB1: Timer B - Output 1</li> <li>– HRTIM_OUTPUT_TB2: Timer B - Output 2</li> <li>– HRTIM_OUTPUT_TC1: Timer C - Output 1</li> <li>– HRTIM_OUTPUT_TC2: Timer C - Output 2</li> <li>– HRTIM_OUTPUT_TD1: Timer D - Output 1</li> <li>– HRTIM_OUTPUT_TD2: Timer D - Output 2</li> <li>– HRTIM_OUTPUT_TE1: Timer E - Output 1</li> <li>– HRTIM_OUTPUT_TE2: Timer E - Output 2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Delayed:</b> protection status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

**HAL\_HRTIM\_GetBurstStatus**

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_HRTIM_GetBurstStatus (HRTIM_HandleTypeDef<br/>* hhrtim)</code>              |
| Function description | Returns the actual status (active or inactive) of the burst mode controller.                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Burst:</b> mode controller status</li> </ul>       |

**HAL\_HRTIM\_GetCurrentPushPullStatus**

|                      |                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_HRTIM_GetCurrentPushPullStatus<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx)</code>                                                                                                                                                                                                                                                                      |
| Function description | Indicates on which output the signal is currently active (when the push pull mode is enabled).                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> </ul> </li> </ul> |

|               |                                                                                                                                          |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>Burst:</b> mode controller status</li> </ul>                                                 |

### HAL\_HRTIM\_GetIdlePushPullStatus

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_HRTIM_GetIdlePushPullStatus (HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx)</code>                                                                                                                                                                                                                                                                                                                                                                             |
| Function description | Indicates on which output the signal was applied, in push-pull mode, balanced fault mode or delayed idle mode, when the protection was triggered.                                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Idle:</b> Push Pull Status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                   |

### HAL\_HRTIM\_IRQHandler

|                      |                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void HAL_HRTIM_IRQHandler (HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx)</code>                                                                                                    |
| Function description | This function handles HRTIM interrupt request.                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be any value of HRTIM Timer Index</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                             |

### HAL\_HRTIM\_Fault1Callback

|                      |                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------|
| Function name        | <code>void HAL_HRTIM_Fault1Callback (HRTIM_HandleTypeDef * hhrtim)</code>                        |
| Function description | Callback function invoked when a fault 1 interrupt occurred.                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle *</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> <li>• <b>None</b></li> </ul>           |

### HAL\_HRTIM\_Fault2Callback

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <code>void HAL_HRTIM_Fault2Callback (HRTIM_HandleTypeDef * hhrtim)</code>                      |
| Function description | Callback function invoked when a fault 2 interrupt occurred.                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |

|               |                                                        |
|---------------|--------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>None</li> </ul> |
|---------------|--------------------------------------------------------|

### **HAL\_HRTIM\_Fault3Callback**

|                      |                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_Fault3Callback (HRTIM_HandleTypeDef * hhrtim)</b>                          |
| Function description | Callback function invoked when a fault 3 interrupt occurred.                                 |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>None</li> </ul>                                       |

### **HAL\_HRTIM\_Fault4Callback**

|                      |                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_Fault4Callback (HRTIM_HandleTypeDef * hhrtim)</b>                          |
| Function description | Callback function invoked when a fault 4 interrupt occurred.                                 |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>None</li> </ul>                                       |

### **HAL\_HRTIM\_Fault5Callback**

|                      |                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_Fault5Callback (HRTIM_HandleTypeDef * hhrtim)</b>                          |
| Function description | Callback function invoked when a fault 5 interrupt occurred.                                 |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>None</li> </ul>                                       |

### **HAL\_HRTIM\_SystemFaultCallback**

|                      |                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_SystemFaultCallback (HRTIM_HandleTypeDef * hhrtim)</b>                     |
| Function description | Callback function invoked when a system fault interrupt occurred.                            |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>None</li> </ul>                                       |

### **HAL\_HRTIM\_DLLCalibrationReadyCallback**

|                      |                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_DLLCalibrationReadyCallback (HRTIM_HandleTypeDef * hhrtim)</b>             |
| Function description | Callback function invoked when the DLL calibration is completed.                             |
| Parameters           | <ul style="list-style-type: none"> <li><b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>None</li> </ul>                                       |

**HAL\_HRTIM\_BurstModePeriodCallback**

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_BurstModePeriodCallback<br/>(HRTIM_HandleTypeDef * hhrtim)</b>               |
| Function description | Callback function invoked when the end of the burst mode period is reached.                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                |

**HAL\_HRTIM\_SynchronizationEventCallback**

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_SynchronizationEventCallback<br/>(HRTIM_HandleTypeDef * hhrtim)</b>          |
| Function description | Callback function invoked when a synchronization input event is received.                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                |

**HAL\_HRTIM\_RegistersUpdateCallback**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_RegistersUpdateCallback<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Callback function invoked when timer registers are updated.                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_MASTER for master timer</li> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>- HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

**HAL\_HRTIM\_RepetitionEventCallback**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_RepetitionEventCallback<br/>(HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx)</b>                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Callback function invoked when timer repetition period has elapsed.                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- HRTIM_TIMERINDEX_MASTER for master timer</li> <li>- HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>- HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>- HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>- HRTIM_TIMERINDEX_TIMER_D for timer D</li> </ul> </li> </ul> |

- HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values • **None**

### **HAL\_HRTIM\_Compare1EventCallback**

Function name **void HAL\_HRTIM\_Compare1EventCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description Callback function invoked when the timer counter matches the value programmed in the compare 1 register.

- Parameters
- **hhrtim:** pointer to HAL HRTIM handle
  - **TimerIdx:** Timer index This parameter can be one of the following values:
    - HRTIM\_TIMERINDEX\_MASTER for master timer
    - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
    - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
    - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
    - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
    - HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values • **None**

### **HAL\_HRTIM\_Compare2EventCallback**

Function name **void HAL\_HRTIM\_Compare2EventCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description Callback function invoked when the timer counter matches the value programmed in the compare 2 register.

- Parameters
- **hhrtim:** pointer to HAL HRTIM handle
  - **TimerIdx:** Timer index This parameter can be one of the following values:
    - HRTIM\_TIMERINDEX\_MASTER for master timer
    - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
    - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
    - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
    - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
    - HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values • **None**

### **HAL\_HRTIM\_Compare3EventCallback**

Function name **void HAL\_HRTIM\_Compare3EventCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description Callback function invoked when the timer counter matches the value programmed in the compare 3 register.

- Parameters
- **hhrtim:** pointer to HAL HRTIM handle
  - **TimerIdx:** Timer index This parameter can be one of the following values:
    - HRTIM\_TIMERINDEX\_MASTER for master timer
    - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
    - HRTIM\_TIMERINDEX\_TIMER\_B for timer B

- HRTIM\_TIMERINDEX\_TIMER\_C for timer C
- HRTIM\_TIMERINDEX\_TIMER\_D for timer D
- HRTIM\_TIMERINDEX\_TIMER\_E for timer E

**Return values**

- **None**

**HAL\_HRTIM\_Compare4EventCallback****Function name**

**void HAL\_HRTIM\_Compare4EventCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

**Function description**

Callback function invoked when the timer counter matches the value programmed in the compare 4 register.

**Parameters**

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_MASTER for master timer
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E

**Return values**

- **None**

**HAL\_HRTIM\_Capture1EventCallback****Function name**

**void HAL\_HRTIM\_Capture1EventCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

**Function description**

Callback function invoked when the timer x capture 1 event occurs.

**Parameters**

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E

**Return values**

- **None**

**HAL\_HRTIM\_Capture2EventCallback****Function name**

**void HAL\_HRTIM\_Capture2EventCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

**Function description**

Callback function invoked when the timer x capture 2 event occurs.

**Parameters**

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B

- HRTIM\_TIMERINDEX\_TIMER\_C for timer C
- HRTIM\_TIMERINDEX\_TIMER\_D for timer D
- HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values

- **None**

**HAL\_HRTIM\_DelayedProtectionCallback**

Function name

**void HAL\_HRTIM\_DelayedProtectionCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description

Callback function invoked when the delayed idle or balanced idle mode is entered.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values

- **None**

**HAL\_HRTIM\_CounterResetCallback**

Function name

**void HAL\_HRTIM\_CounterResetCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description

Callback function invoked when the timer x counter reset/roll-over event occurs.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values

- **None**

**HAL\_HRTIM\_Output1SetCallback**

Function name

**void HAL\_HRTIM\_Output1SetCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description

Callback function invoked when the timer x output 1 is set.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D

- HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values

- **None**

### **HAL\_HRTIM\_Output1ResetCallback**

Function name

**void HAL\_HRTIM\_Output1ResetCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description

Callback function invoked when the timer x output 1 is reset.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values

- **None**

### **HAL\_HRTIM\_Output2SetCallback**

Function name

**void HAL\_HRTIM\_Output2SetCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description

Callback function invoked when the timer x output 2 is set.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values

- **None**

### **HAL\_HRTIM\_Output2ResetCallback**

Function name

**void HAL\_HRTIM\_Output2ResetCallback  
(HRTIM\_HandleTypeDef \* hhrtim, uint32\_t TimerIdx)**

Function description

Callback function invoked when the timer x output 2 is reset.

Parameters

- **hhrtim:** pointer to HAL HRTIM handle
- **TimerIdx:** Timer index This parameter can be one of the following values:
  - HRTIM\_TIMERINDEX\_TIMER\_A for timer A
  - HRTIM\_TIMERINDEX\_TIMER\_B for timer B
  - HRTIM\_TIMERINDEX\_TIMER\_C for timer C
  - HRTIM\_TIMERINDEX\_TIMER\_D for timer D
  - HRTIM\_TIMERINDEX\_TIMER\_E for timer E

Return values

- **None**

**HAL\_HRTIM\_BurstDMATransferCallback**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_BurstDMATransferCallback (HRTIM_HandleTypeDef * hhrtim, uint32_t TimerIdx)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Callback function invoked when a DMA burst transfer is completed.                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> <li>• <b>TimerIdx:</b> Timer index This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– HRTIM_TIMERINDEX_MASTER for master timer</li> <li>– HRTIM_TIMERINDEX_TIMER_A for timer A</li> <li>– HRTIM_TIMERINDEX_TIMER_B for timer B</li> <li>– HRTIM_TIMERINDEX_TIMER_C for timer C</li> <li>– HRTIM_TIMERINDEX_TIMER_D for timer D</li> <li>– HRTIM_TIMERINDEX_TIMER_E for timer E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

**HAL\_HRTIM\_ErrorCallback**

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_HRTIM_ErrorCallback (HRTIM_HandleTypeDef * hhrtim)</b>                             |
| Function description | Callback function invoked when a DMA error occurs.                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hhrtim:</b> pointer to HAL HRTIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                |

## 23.3 HRTIM Firmware driver defines

### 23.3.1 HRTIM

***HRTIM ADC Trigger***

|                            |                          |
|----------------------------|--------------------------|
| HRTIM_ADCTRIGGER_1         | ADC trigger 1 identifier |
| HRTIM_ADCTRIGGER_2         | ADC trigger 2 identifier |
| HRTIM_ADCTRIGGER_3         | ADC trigger 3 identifier |
| HRTIM_ADCTRIGGER_4         | ADC trigger 4 identifier |
| <b>IS_HRTIM_ADCTRIGGER</b> |                          |

***HRTIM ADC Trigger Event***

|                                     |                                  |
|-------------------------------------|----------------------------------|
| HRTIM_ADCTRIGGEREVENT13_NONE        | No ADC trigger event             |
| HRTIM_ADCTRIGGEREVENT13_MASTER_CMP1 | ADC Trigger on master compare 1U |
| HRTIM_ADCTRIGGEREVENT13_MASTER_CMP2 | ADC Trigger on master compare 2U |
| HRTIM_ADCTRIGGEREVENT13_MASTER_CMP3 | ADC Trigger on master compare 3U |
| HRTIM_ADCTRIGGEREVENT13_MASTER_CMP4 | ADC Trigger on master compare 4U |

|                                       |                                   |
|---------------------------------------|-----------------------------------|
| HRTIM_ADCTRIGGEREVENT13_MASTER_PERIOD | ADC Trigger on master period      |
| HRTIM_ADCTRIGGEREVENT13_EVENT_1       | ADC Trigger on external event 1U  |
| HRTIM_ADCTRIGGEREVENT13_EVENT_2       | ADC Trigger on external event 2U  |
| HRTIM_ADCTRIGGEREVENT13_EVENT_3       | ADC Trigger on external event 3U  |
| HRTIM_ADCTRIGGEREVENT13_EVENT_4       | ADC Trigger on external event 4U  |
| HRTIM_ADCTRIGGEREVENT13_EVENT_5       | ADC Trigger on external event 5U  |
| HRTIM_ADCTRIGGEREVENT13_TIMERA_CMP2   | ADC Trigger on Timer A compare 2U |
| HRTIM_ADCTRIGGEREVENT13_TIMERA_CMP3   | ADC Trigger on Timer A compare 3U |
| HRTIM_ADCTRIGGEREVENT13_TIMERA_CMP4   | ADC Trigger on Timer A compare 4U |
| HRTIM_ADCTRIGGEREVENT13_TIMERA_PERIOD | ADC Trigger on Timer A period     |
| HRTIM_ADCTRIGGEREVENT13_TIMERA_RESET  | ADC Trigger on Timer A reset      |
| HRTIM_ADCTRIGGEREVENT13_TIMERB_CMP2   | ADC Trigger on Timer B compare 2U |
| HRTIM_ADCTRIGGEREVENT13_TIMERB_CMP3   | ADC Trigger on Timer B compare 3U |
| HRTIM_ADCTRIGGEREVENT13_TIMERB_CMP4   | ADC Trigger on Timer B compare 4U |
| HRTIM_ADCTRIGGEREVENT13_TIMERB_PERIOD | ADC Trigger on Timer B period     |
| HRTIM_ADCTRIGGEREVENT13_TIMERB_RESET  | ADC Trigger on Timer B reset      |
| HRTIM_ADCTRIGGEREVENT13_TIMERC_CMP2   | ADC Trigger on Timer C compare 2U |
| HRTIM_ADCTRIGGEREVENT13_TIMERC_CMP3   | ADC Trigger on Timer C compare 3U |
| HRTIM_ADCTRIGGEREVENT13_TIMERC_CMP4   | ADC Trigger on Timer C compare 4U |
| HRTIM_ADCTRIGGEREVENT13_TIMERC_PERIOD | ADC Trigger on Timer C period     |
| HRTIM_ADCTRIGGEREVENT13_TIMERD_CMP2   | ADC Trigger on Timer D compare 2U |
| HRTIM_ADCTRIGGEREVENT13_TIMERD_CMP3   | ADC Trigger on Timer D compare 3U |
| HRTIM_ADCTRIGGEREVENT13_TIMERD_CMP4   | ADC Trigger on Timer D compare 4U |
| HRTIM_ADCTRIGGEREVENT13_TIMERD_PERIOD | ADC Trigger on Timer D period     |
| HRTIM_ADCTRIGGEREVENT13_TIMERE_CMP2   | ADC Trigger on Timer E compare 2U |
| HRTIM_ADCTRIGGEREVENT13_TIMERE_CMP3   | ADC Trigger on Timer E compare 3U |
| HRTIM_ADCTRIGGEREVENT13_TIMERE_CMP4   | ADC Trigger on Timer E compare    |

|                                       |                                   |
|---------------------------------------|-----------------------------------|
|                                       | 4U                                |
| HRTIM_ADCTRIGGEREVENT13_TIMERE_PERIOD | ADC Trigger on Timer E period     |
| HRTIM_ADCTRIGGEREVENT24_NONE          | No ADC trigger event              |
| HRTIM_ADCTRIGGEREVENT24_MASTER_CMP1   | ADC Trigger on master compare 1U  |
| HRTIM_ADCTRIGGEREVENT24_MASTER_CMP2   | ADC Trigger on master compare 2U  |
| HRTIM_ADCTRIGGEREVENT24_MASTER_CMP3   | ADC Trigger on master compare 3U  |
| HRTIM_ADCTRIGGEREVENT24_MASTER_CMP4   | ADC Trigger on master compare 4U  |
| HRTIM_ADCTRIGGEREVENT24_MASTER_PERIOD | ADC Trigger on master period      |
| HRTIM_ADCTRIGGEREVENT24_EVENT_6       | ADC Trigger on external event 6U  |
| HRTIM_ADCTRIGGEREVENT24_EVENT_7       | ADC Trigger on external event 7U  |
| HRTIM_ADCTRIGGEREVENT24_EVENT_8       | ADC Trigger on external event 8U  |
| HRTIM_ADCTRIGGEREVENT24_EVENT_9       | ADC Trigger on external event 9U  |
| HRTIM_ADCTRIGGEREVENT24_EVENT_10      | ADC Trigger on external event 10U |
| HRTIM_ADCTRIGGEREVENT24_TIMERA_CMP2   | ADC Trigger on Timer A compare 2U |
| HRTIM_ADCTRIGGEREVENT24_TIMERA_CMP3   | ADC Trigger on Timer A compare 3U |
| HRTIM_ADCTRIGGEREVENT24_TIMERA_CMP4   | ADC Trigger on Timer A compare 4U |
| HRTIM_ADCTRIGGEREVENT24_TIMERA_PERIOD | ADC Trigger on Timer A period     |
| HRTIM_ADCTRIGGEREVENT24_TIMERB_CMP2   | ADC Trigger on Timer B compare 2U |
| HRTIM_ADCTRIGGEREVENT24_TIMERB_CMP3   | ADC Trigger on Timer B compare 3U |
| HRTIM_ADCTRIGGEREVENT24_TIMERB_CMP4   | ADC Trigger on Timer B compare 4U |
| HRTIM_ADCTRIGGEREVENT24_TIMERB_PERIOD | ADC Trigger on Timer B period     |
| HRTIM_ADCTRIGGEREVENT24_TIMERC_CMP2   | ADC Trigger on Timer C compare 2U |
| HRTIM_ADCTRIGGEREVENT24_TIMERC_CMP3   | ADC Trigger on Timer C compare 3U |
| HRTIM_ADCTRIGGEREVENT24_TIMERC_CMP4   | ADC Trigger on Timer C compare 4U |
| HRTIM_ADCTRIGGEREVENT24_TIMERC_PERIOD | ADC Trigger on Timer C period     |
| HRTIM_ADCTRIGGEREVENT24_TIMERC_RESET  | ADC Trigger on Timer C reset      |
| HRTIM_ADCTRIGGEREVENT24_TIMERD_CMP2   | ADC Trigger on Timer D compare    |

|                                       |                                   |
|---------------------------------------|-----------------------------------|
|                                       | 2U                                |
| HRTIM_ADCTRIGGEREVENT24_TIMERD_CMP3   | ADC Trigger on Timer D compare 3U |
| HRTIM_ADCTRIGGEREVENT24_TIMERD_CMP4   | ADC Trigger on Timer D compare 4U |
| HRTIM_ADCTRIGGEREVENT24_TIMERD_PERIOD | ADC Trigger on Timer D period     |
| HRTIM_ADCTRIGGEREVENT24_TIMERD_RESET  | ADC Trigger on Timer D reset      |
| HRTIM_ADCTRIGGEREVENT24_TIMERE_CMP2   | ADC Trigger on Timer E compare 2U |
| HRTIM_ADCTRIGGEREVENT24_TIMERE_CMP3   | ADC Trigger on Timer E compare 3U |
| HRTIM_ADCTRIGGEREVENT24_TIMERE_CMP4   | ADC Trigger on Timer E compare 4U |
| HRTIM_ADCTRIGGEREVENT24_TIMERE_RESET  | ADC Trigger on Timer E reset      |

***HRTIM ADC Trigger Update Source***

|                                |              |
|--------------------------------|--------------|
| HRTIM_ADCTRIGGERUPDATE_MASTER  | Master timer |
| HRTIM_ADCTRIGGERUPDATE_TIMER_A | Timer A      |
| HRTIM_ADCTRIGGERUPDATE_TIMER_B | Timer B      |
| HRTIM_ADCTRIGGERUPDATE_TIMER_C | Timer C      |
| HRTIM_ADCTRIGGERUPDATE_TIMER_D | Timer D      |
| HRTIM_ADCTRIGGERUPDATE_TIMER_E | Timer E      |

***HRTIM Burst DMA Registers Update***

|                     |                                                             |
|---------------------|-------------------------------------------------------------|
| HRTIM_BURSTDMA_NONE | No register is updated by Burst DMA accesses                |
| HRTIM_BURSTDMA_CR   | MCR or TIMxCR register is updated by Burst DMA accesses     |
| HRTIM_BURSTDMA_ICR  | MICR or TIMxICR register is updated by Burst DMA accesses   |
| HRTIM_BURSTDMA_DIER | MDIER or TIMxDIER register is updated by Burst DMA accesses |
| HRTIM_BURSTDMA_CNT  | MCNTR or CNTxCR register is updated by Burst DMA accesses   |
| HRTIM_BURSTDMA_PER  | MPER or PERxR register is updated by Burst DMA accesses     |
| HRTIM_BURSTDMA REP  | MREPR or REPxR register is updated by Burst DMA accesses    |
| HRTIM_BURSTDMA_CMP1 | MCMP1R or CMP1xR register is updated by Burst DMA accesses  |
| HRTIM_BURSTDMA_CMP2 | MCMP2R or CMP2xR register is updated by Burst DMA accesses  |
| HRTIM_BURSTDMA_CMP3 | MCMP3R or CMP3xR register is updated by Burst DMA accesses  |

|                                      |                                                                                         |
|--------------------------------------|-----------------------------------------------------------------------------------------|
| HRTIM_BURSTDMA_CMP4                  | MCMP4R or CMP4xR register is updated by Burst DMA accesses                              |
| HRTIM_BURSTDMA_DTR                   | TDxR register is updated by Burst DMA accesses                                          |
| HRTIM_BURSTDMA_SET1R                 | SET1R register is updated by Burst DMA accesses                                         |
| HRTIM_BURSTDMA_RST1R                 | RST1R register is updated by Burst DMA accesses                                         |
| HRTIM_BURSTDMA_SET2R                 | SET2R register is updated by Burst DMA accesses                                         |
| HRTIM_BURSTDMA_RST2R                 | RST1R register is updated by Burst DMA accesses                                         |
| HRTIM_BURSTDMA_EEFR1                 | EEFxR1 register is updated by Burst DMA accesses                                        |
| HRTIM_BURSTDMA_EEFR2                 | EEFxR2 register is updated by Burst DMA accesses                                        |
| HRTIM_BURSTDMA_RSTR                  | RSTxR register is updated by Burst DMA accesses                                         |
| HRTIM_BURSTDMA_CHPR                  | CHPxR register is updated by Burst DMA accesses                                         |
| HRTIM_BURSTDMA_OUTR                  | OUTxR register is updated by Burst DMA accesses                                         |
| HRTIM_BURSTDMA_FLTR                  | FLTxD register is updated by Burst DMA accesses                                         |
| <b>HRTIM Burst Mode Clock Source</b> |                                                                                         |
| HRTIM_BURSTMODECLOCKSOURCE_MASTER    | Master timer counter reset/roll-over is used as clock source for the burst mode counter |
| HRTIM_BURSTMODECLOCKSOURCE_TIMER_A   | Timer A counter reset/roll-over is used as clock source for the burst mode counter      |
| HRTIM_BURSTMODECLOCKSOURCE_TIMER_B   | Timer B counter reset/roll-over is used as clock source for the burst mode counter      |
| HRTIM_BURSTMODECLOCKSOURCE_TIMER_C   | Timer C counter reset/roll-over is used as clock source for the burst mode counter      |
| HRTIM_BURSTMODECLOCKSOURCE_TIMER_D   | Timer D counter reset/roll-over is used as clock source for the burst mode counter      |
| HRTIM_BURSTMODECLOCKSOURCE_TIMER_E   | Timer E counter reset/roll-over is used as clock source for the burst mode counter      |
| HRTIM_BURSTMODECLOCKSOURCE_TIM16_OC  | On-chip Event 1 (BMClk[1]), acting as a burst mode counter clock                        |
| HRTIM_BURSTMODECLOCKSOURCE_TIM17_OC  | On-chip Event 2 (BMClk[2]), acting as a burst mode counter clock                        |
| HRTIM_BURSTMODECLOCKSOURCE_TIM7_TRGO | On-chip Event 3 (BMClk[3]), acting as a burst mode counter clock                        |
| HRTIM_BURSTMODECLOCKSOURCE_FHRTIM    | Prescaled fHRTIM clock is used as clock source for the burst mode counter               |

***HRTIM Burst Mode Control***

HRTIM\_BURSTMODECTL\_DISABLED      Burst mode disabled

HRTIM\_BURSTMODECTL\_ENABLED      Burst mode enabled

***HRTIM Burst Mode Operating Mode***

HRTIM\_BURSTMODE\_SINGLESHOT      Burst mode operates in single shot mode

HRTIM\_BURSTMODE\_CONTINOUS      Burst mode operates in continuous mode

***HRTIM Burst Mode Prescaler***

HRTIM\_BURSTMODEPRESCALER\_DIV1       $f_{BRST} = f_{HRTIM}$

HRTIM\_BURSTMODEPRESCALER\_DIV2       $f_{BRST} = f_{HRTIM}/2U$

HRTIM\_BURSTMODEPRESCALER\_DIV4       $f_{BRST} = f_{HRTIM}/4U$

HRTIM\_BURSTMODEPRESCALER\_DIV8       $f_{BRST} = f_{HRTIM}/8U$

HRTIM\_BURSTMODEPRESCALER\_DIV16       $f_{BRST} = f_{HRTIM}/16U$

HRTIM\_BURSTMODEPRESCALER\_DIV32       $f_{BRST} = f_{HRTIM}/32U$

HRTIM\_BURSTMODEPRESCALER\_DIV64       $f_{BRST} = f_{HRTIM}/64U$

HRTIM\_BURSTMODEPRESCALER\_DIV128       $f_{BRST} = f_{HRTIM}/128U$

HRTIM\_BURSTMODEPRESCALER\_DIV256       $f_{BRST} = f_{HRTIM}/256U$

HRTIM\_BURSTMODEPRESCALER\_DIV512       $f_{BRST} = f_{HRTIM}/512U$

HRTIM\_BURSTMODEPRESCALER\_DIV1024       $f_{BRST} = f_{HRTIM}/1024U$

HRTIM\_BURSTMODEPRESCALER\_DIV2048       $f_{BRST} = f_{HRTIM}/2048U$

HRTIM\_BURSTMODEPRESCALER\_DIV4096       $f_{BRST} = f_{HRTIM}/4096U$

HRTIM\_BURSTMODEPRESCALER\_DIV8192       $f_{BRST} = f_{HRTIM}/8192U$

HRTIM\_BURSTMODEPRESCALER\_DIV16384       $f_{BRST} = f_{HRTIM}/16384U$

HRTIM\_BURSTMODEPRESCALER\_DIV32768       $f_{BRST} = f_{HRTIM}/32768U$

***HRTIM Burst Mode Register Preload Enable***

HRIM\_BURSTMODEPRELOAD\_DISABLED      Preload disabled: the write access is directly done into active registers

HRIM\_BURSTMODEPRELOAD\_ENABLED      Preload enabled: the write access is done into preload registers

***HRTIM Burst Mode Status***

HRTIM\_BURSTMODESTATUS\_NORMAL      Normal operation

HRTIM\_BURSTMODESTATUS\_ONGOING      Burst operation on-going

***HRTIM Burst Mode Trigger***

HRTIM\_BURSTMODETRIGGER\_NONE      No trigger

HRTIM\_BURSTMODETRIGGER\_MASTER\_RESET      Master reset

HRTIM\_BURSTMODETRIGGER\_MASTER\_REPETITION      Master repetition

HRTIM\_BURSTMODETRIGGER\_MASTER\_CMP1      Master compare 1U

HRTIM\_BURSTMODETRIGGER\_MASTER\_CMP2      Master compare 2U

|                                          |                                            |
|------------------------------------------|--------------------------------------------|
| HRTIM_BURSTMODETRIGGER_MASTER_CMP3       | Master compare 3U                          |
| HRTIM_BURSTMODETRIGGER_MASTER_CMP4       | Master compare 4U                          |
| HRTIM_BURSTMODETRIGGER_TIMERA_RESET      | Timer A reset                              |
| HRTIM_BURSTMODETRIGGER_TIMERA_REPETITION | Timer A repetition                         |
| HRTIM_BURSTMODETRIGGER_TIMERA_CMP1       | Timer A compare 1                          |
| HRTIM_BURSTMODETRIGGER_TIMERA_CMP2       | Timer A compare 2                          |
| HRTIM_BURSTMODETRIGGER_TIMERB_RESET      | Timer B reset                              |
| HRTIM_BURSTMODETRIGGER_TIMERB_REPETITION | Timer B repetition                         |
| HRTIM_BURSTMODETRIGGER_TIMERB_CMP1       | Timer B compare 1                          |
| HRTIM_BURSTMODETRIGGER_TIMERB_CMP2       | Timer B compare 2                          |
| HRTIM_BURSTMODETRIGGER_TIMERC_RESET      | Timer C reset                              |
| HRTIM_BURSTMODETRIGGER_TIMERC_REPETITION | Timer C repetition                         |
| HRTIM_BURSTMODETRIGGER_TIMERC_CMP1       | Timer C compare 1                          |
| HRTIM_BURSTMODETRIGGER_TIMERC_CMP2       | Timer C compare 2                          |
| HRTIM_BURSTMODETRIGGER_TIMERD_RESET      | Timer D reset                              |
| HRTIM_BURSTMODETRIGGER_TIMERD_REPETITION | Timer D repetition                         |
| HRTIM_BURSTMODETRIGGER_TIMERD_CMP1       | Timer D compare 1                          |
| HRTIM_BURSTMODETRIGGER_TIMERD_CMP2       | Timer D compare 2                          |
| HRTIM_BURSTMODETRIGGER_TIMERE_RESET      | Timer E reset                              |
| HRTIM_BURSTMODETRIGGER_TIMERE_REPETITION | Timer E repetition                         |
| HRTIM_BURSTMODETRIGGER_TIMERE_CMP1       | Timer E compare 1                          |
| HRTIM_BURSTMODETRIGGER_TIMERE_CMP2       | Timer E compare 2                          |
| HRTIM_BURSTMODETRIGGER_TIMERA_EVENT7     | Timer A period following External Event 7  |
| HRTIM_BURSTMODETRIGGER_TIMERD_EVENT8     | Timer D period following External Event 8  |
| HRTIM_BURSTMODETRIGGER_EVENT_7           | External Event 7 (timer A filters applied) |
| HRTIM_BURSTMODETRIGGER_EVENT_8           | External Event 8 (timer D filters applied) |
| HRTIM_BURSTMODETRIGGER_EVENT_ONCHIP      | On-chip Event                              |
| <b>HRTIM Capture Unit</b>                |                                            |
| HRTIM_CAPTUREUNIT_1                      | Capture unit 1 identifier                  |
| HRTIM_CAPTUREUNIT_2                      | Capture unit 2 identifier                  |
| <b>HRTIM Capture Unit Trigger</b>        |                                            |
| HRTIM_CAPTURETRIGGER_NONE                | Capture trigger is disabled                |
| HRTIM_CAPTURETRIGGER_UPDATE              | The update event triggers the Capture      |
| HRTIM_CAPTURETRIGGER_EEV_1               | The External event 1 triggers the Capture  |

|                                  |                                                                  |
|----------------------------------|------------------------------------------------------------------|
| HRTIM_CAPTURETRIGGER_EEV_2       | The External event 2 triggers the Capture                        |
| HRTIM_CAPTURETRIGGER_EEV_3       | The External event 3 triggers the Capture                        |
| HRTIM_CAPTURETRIGGER_EEV_4       | The External event 4 triggers the Capture                        |
| HRTIM_CAPTURETRIGGER_EEV_5       | The External event 5 triggers the Capture                        |
| HRTIM_CAPTURETRIGGER_EEV_6       | The External event 6 triggers the Capture                        |
| HRTIM_CAPTURETRIGGER_EEV_7       | The External event 7 triggers the Capture                        |
| HRTIM_CAPTURETRIGGER_EEV_8       | The External event 8 triggers the Capture                        |
| HRTIM_CAPTURETRIGGER_EEV_9       | The External event 9 triggers the Capture                        |
| HRTIM_CAPTURETRIGGER_EEV_10      | The External event 10 triggers the Capture                       |
| HRTIM_CAPTURETRIGGER_TA1_SET     | Capture is triggered by TA1 output inactive to active transition |
| HRTIM_CAPTURETRIGGER_TA1_RESET   | Capture is triggered by TA1 output active to inactive transition |
| HRTIM_CAPTURETRIGGER_TIMERA_CMP1 | Timer A Compare 1 triggers Capture                               |
| HRTIM_CAPTURETRIGGER_TIMERA_CMP2 | Timer A Compare 2 triggers Capture                               |
| HRTIM_CAPTURETRIGGER_TB1_SET     | Capture is triggered by TB1 output inactive to active transition |
| HRTIM_CAPTURETRIGGER_TB1_RESET   | Capture is triggered by TB1 output active to inactive transition |
| HRTIM_CAPTURETRIGGER_TIMERB_CMP1 | Timer B Compare 1 triggers Capture                               |
| HRTIM_CAPTURETRIGGER_TIMERB_CMP2 | Timer B Compare 2 triggers Capture                               |
| HRTIM_CAPTURETRIGGER_TC1_SET     | Capture is triggered by TC1 output inactive to active transition |
| HRTIM_CAPTURETRIGGER_TC1_RESET   | Capture is triggered by TC1 output active to inactive transition |
| HRTIM_CAPTURETRIGGER_TIMERC_CMP1 | Timer C Compare 1 triggers Capture                               |
| HRTIM_CAPTURETRIGGER_TIMERC_CMP2 | Timer C Compare 2 triggers Capture                               |
| HRTIM_CAPTURETRIGGER_TD1_SET     | Capture is triggered by TD1 output inactive to active transition |
| HRTIM_CAPTURETRIGGER_TD1_RESET   | Capture is triggered by TD1 output active to inactive transition |
| HRTIM_CAPTURETRIGGER_TIMERD_CMP1 | Timer D Compare 1 triggers Capture                               |
| HRTIM_CAPTURETRIGGER_TIMERD_CMP2 | Timer D Compare 2 triggers Capture                               |
| HRTIM_CAPTURETRIGGER_TE1_SET     | Capture is triggered by TE1 output inactive to active transition |
| HRTIM_CAPTURETRIGGER_TE1_RESET   | Capture is triggered by TE1 output active to inactive transition |
| HRTIM_CAPTURETRIGGER_TIMERE_CMP1 | Timer E Compare 1 triggers Capture                               |
| HRTIM_CAPTURETRIGGER_TIMERE_CMP2 | Timer E Compare 2 triggers Capture                               |

***HRTIM Chopper Duty Cycle***

|                             |                                             |
|-----------------------------|---------------------------------------------|
| HRTIM_CHOPPER_DUTYCYCLE_0   | Only 1st pulse is present                   |
| HRTIM_CHOPPER_DUTYCYCLE_125 | Duty cycle of the carrier signal is 12.5U % |
| HRTIM_CHOPPER_DUTYCYCLE_250 | Duty cycle of the carrier signal is 25U %   |
| HRTIM_CHOPPER_DUTYCYCLE_375 | Duty cycle of the carrier signal is 37.5U % |
| HRTIM_CHOPPER_DUTYCYCLE_500 | Duty cycle of the carrier signal is 50U %   |
| HRTIM_CHOPPER_DUTYCYCLE_625 | Duty cycle of the carrier signal is 62.5U % |
| HRTIM_CHOPPER_DUTYCYCLE_750 | Duty cycle of the carrier signal is 75U %   |
| HRTIM_CHOPPER_DUTYCYCLE_875 | Duty cycle of the carrier signal is 87.5U % |

***HRTIM Chopper Frequency***

|                                     |                                |
|-------------------------------------|--------------------------------|
| HRTIM_CHOPPER_PRESCALERRATIO_DIV16  | $f_{CHPFRQ} = f_{HRTIM} / 16$  |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV32  | $f_{CHPFRQ} = f_{HRTIM} / 32$  |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV48  | $f_{CHPFRQ} = f_{HRTIM} / 48$  |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV64  | $f_{CHPFRQ} = f_{HRTIM} / 64$  |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV80  | $f_{CHPFRQ} = f_{HRTIM} / 80$  |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV96  | $f_{CHPFRQ} = f_{HRTIM} / 96$  |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV112 | $f_{CHPFRQ} = f_{HRTIM} / 112$ |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV128 | $f_{CHPFRQ} = f_{HRTIM} / 128$ |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV144 | $f_{CHPFRQ} = f_{HRTIM} / 144$ |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV160 | $f_{CHPFRQ} = f_{HRTIM} / 160$ |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV176 | $f_{CHPFRQ} = f_{HRTIM} / 176$ |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV192 | $f_{CHPFRQ} = f_{HRTIM} / 192$ |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV208 | $f_{CHPFRQ} = f_{HRTIM} / 208$ |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV224 | $f_{CHPFRQ} = f_{HRTIM} / 224$ |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV240 | $f_{CHPFRQ} = f_{HRTIM} / 240$ |
| HRTIM_CHOPPER_PRESCALERRATIO_DIV256 | $f_{CHPFRQ} = f_{HRTIM} / 256$ |

***HRTIM Chopper Start Pulse Width***

|                              |                                   |
|------------------------------|-----------------------------------|
| HRTIM_CHOPPER_PULSEWIDTH_16  | $t_{STPW} = t_{HRTIM} \times 16$  |
| HRTIM_CHOPPER_PULSEWIDTH_32  | $t_{STPW} = t_{HRTIM} \times 32$  |
| HRTIM_CHOPPER_PULSEWIDTH_48  | $t_{STPW} = t_{HRTIM} \times 48$  |
| HRTIM_CHOPPER_PULSEWIDTH_64  | $t_{STPW} = t_{HRTIM} \times 64$  |
| HRTIM_CHOPPER_PULSEWIDTH_80  | $t_{STPW} = t_{HRTIM} \times 80$  |
| HRTIM_CHOPPER_PULSEWIDTH_96  | $t_{STPW} = t_{HRTIM} \times 96$  |
| HRTIM_CHOPPER_PULSEWIDTH_112 | $t_{STPW} = t_{HRTIM} \times 112$ |
| HRTIM_CHOPPER_PULSEWIDTH_128 | $t_{STPW} = t_{HRTIM} \times 128$ |
| HRTIM_CHOPPER_PULSEWIDTH_144 | $t_{STPW} = t_{HRTIM} \times 144$ |

|                              |                                   |
|------------------------------|-----------------------------------|
| HRTIM_CHOPPER_PULSEWIDTH_160 | $t_{STPW} = t_{HRTIM} \times 160$ |
| HRTIM_CHOPPER_PULSEWIDTH_176 | $t_{STPW} = t_{HRTIM} \times 176$ |
| HRTIM_CHOPPER_PULSEWIDTH_192 | $t_{STPW} = t_{HRTIM} \times 192$ |
| HRTIM_CHOPPER_PULSEWIDTH_208 | $t_{STPW} = t_{HRTIM} \times 208$ |
| HRTIM_CHOPPER_PULSEWIDTH_224 | $t_{STPW} = t_{HRTIM} \times 224$ |
| HRTIM_CHOPPER_PULSEWIDTH_240 | $t_{STPW} = t_{HRTIM} \times 240$ |
| HRTIM_CHOPPER_PULSEWIDTH_256 | $t_{STPW} = t_{HRTIM} \times 256$ |

***HRTIM Common Interrupt Enable***

|                 |                                    |
|-----------------|------------------------------------|
| HRTIM_IT_NONE   | No interrupt enabled               |
| HRTIM_IT_FLT1   | Fault 1 interrupt enable           |
| HRTIM_IT_FLT2   | Fault 2 interrupt enable           |
| HRTIM_IT_FLT3   | Fault 3 interrupt enable           |
| HRTIM_IT_FLT4   | Fault 4 interrupt enable           |
| HRTIM_IT_FLT5   | Fault 5 interrupt enable           |
| HRTIM_IT_SYSFLT | System Fault interrupt enable      |
| HRTIM_IT_DLLRDY | DLL ready interrupt enable         |
| HRTIM_IT_BMPER  | Burst mode period interrupt enable |

***HRTIM Common Interrupt Flag***

|                   |                                  |
|-------------------|----------------------------------|
| HRTIM_FLAG_FLT1   | Fault 1 interrupt flag           |
| HRTIM_FLAG_FLT2   | Fault 2 interrupt flag           |
| HRTIM_FLAG_FLT3   | Fault 3 interrupt flag           |
| HRTIM_FLAG_FLT4   | Fault 4 interrupt flag           |
| HRTIM_FLAG_FLT5   | Fault 5 interrupt flag           |
| HRTIM_FLAG_SYSFLT | System Fault interrupt flag      |
| HRTIM_FLAG_DLLRDY | DLL ready interrupt flag         |
| HRTIM_FLAG_BMPER  | Burst mode period interrupt flag |

***HRTIM Compare Unit***

|                     |                           |
|---------------------|---------------------------|
| HRTIM_COMPAREUNIT_1 | Compare unit 1 identifier |
| HRTIM_COMPAREUNIT_2 | Compare unit 2 identifier |
| HRTIM_COMPAREUNIT_3 | Compare unit 3 identifier |
| HRTIM_COMPAREUNIT_4 | Compare unit 4 identifier |

***HRTIM Compare Unit Auto Delayed Mode***

|                                               |                                                                                          |
|-----------------------------------------------|------------------------------------------------------------------------------------------|
| HRTIM_AUTODELAYEDMODE_REGULAR                 | standard compare mode                                                                    |
| HRTIM_AUTODELAYEDMODE_AUTODELAYED_NOTIMEOUT   | Compare event generated only if a capture has occurred                                   |
| HRTIM_AUTODELAYEDMODE_AUTODELAYED_TIMEOUTCMP1 | Compare event generated if a capture has occurred or after a Compare 1 match (timeout if |

|                                                |                                                                                                                    |
|------------------------------------------------|--------------------------------------------------------------------------------------------------------------------|
|                                                | capture event is missing                                                                                           |
| HRTIM_AUTODELAYEDMODE_AUTODELAYED_TIMEOUTCMP3  | Compare event generated if a capture has occurred or after a Compare 3 match (timeout if capture event is missing) |
| <b><i>HRTIM Counter Operating Mode</i></b>     |                                                                                                                    |
| HRTIM_MODE_CONTINUOUS                          | The timer operates in continuous (free-running) mode                                                               |
| HRTIM_MODE_SINGLESHOT                          | The timer operates in non retriggerable single-shot mode                                                           |
| HRTIM_MODE_SINGLESHOT_RETRIGGERABLE            | The timer operates in retriggerable single-shot mode                                                               |
| <b><i>HRTIM Current Push Pull Status</i></b>   |                                                                                                                    |
| HRTIM_PUSHPULL_CURRENTSTATUS_OUTPUT1           | Signal applied on output 1 and output 2 forced inactive                                                            |
| HRTIM_PUSHPULL_CURRENTSTATUS_OUTPUT2           | Signal applied on output 2 and output 1 forced inactive                                                            |
| <b><i>HRTIM DAC Synchronization</i></b>        |                                                                                                                    |
| HRTIM_DACSYNC_NONE                             | No DAC synchronization event generated                                                                             |
| HRTIM_DACSYNC_DACTRIGOUT_1                     | DAC synchronization event generated on DACTrigOut1 output upon timer update                                        |
| HRTIM_DACSYNC_DACTRIGOUT_2                     | DAC synchronization event generated on DACTrigOut2 output upon timer update                                        |
| HRTIM_DACSYNC_DACTRIGOUT_3                     | DAC update generated on DACTrigOut3 output upon timer update                                                       |
| <b><i>HRTIM Deadtime Falling Lock</i></b>      |                                                                                                                    |
| HRTIM_TIMDEADTIME_FALLINGLOCK_WRITE            | Deadtime falling value and sign is writeable                                                                       |
| HRTIM_TIMDEADTIME_FALLINGLOCK_READONLY         | Deadtime falling value and sign is read-only                                                                       |
| <b><i>HRTIM Deadtime Falling Sign</i></b>      |                                                                                                                    |
| HRTIM_TIMDEADTIME_FALLINGSIGN_POSITIVE         | Positive deadtime on falling edge                                                                                  |
| HRTIM_TIMDEADTIME_FALLINGSIGN_NEGATIVE         | Negative deadtime on falling edge                                                                                  |
| <b><i>HRTIM Deadtime Falling Sign Lock</i></b> |                                                                                                                    |
| HRTIM_TIMDEADTIME_FALLINGSIGNLOCK_WRITE        | Deadtime falling sign is writeable                                                                                 |
| HRTIM_TIMDEADTIME_FALLINGSIGNLOCK_READONLY     | Deadtime falling sign is read-only                                                                                 |
| <b><i>HRTIM Deadtime Prescaler Ratio</i></b>   |                                                                                                                    |
| HRTIM_TIMDEADTIME_PRESCALERRATIO_MUL8          | $fDTG = fHRTIM * 8U$                                                                                               |

|                                        |                       |
|----------------------------------------|-----------------------|
| HRTIM_TIMDEADTIME_PRESCALERRATIO_MUL4  | $fDTG = fHRTIM * 4U$  |
| HRTIM_TIMDEADTIME_PRESCALERRATIO_MUL2  | $fDTG = fHRTIM * 2U$  |
| HRTIM_TIMDEADTIME_PRESCALERRATIO_DIV1  | $fDTG = fHRTIM$       |
| HRTIM_TIMDEADTIME_PRESCALERRATIO_DIV2  | $fDTG = fHRTIM / 2U$  |
| HRTIM_TIMDEADTIME_PRESCALERRATIO_DIV4  | $fDTG = fHRTIM / 4U$  |
| HRTIM_TIMDEADTIME_PRESCALERRATIO_DIV8  | $fDTG = fHRTIM / 8U$  |
| HRTIM_TIMDEADTIME_PRESCALERRATIO_DIV16 | $fDTG = fHRTIM / 16U$ |

***HRTIM Deadtime Rising Lock***

|                                       |                                             |
|---------------------------------------|---------------------------------------------|
| HRTIM_TIMDEADTIME_RISINGLOCK_WRITE    | Deadtime rising value and sign is writeable |
| HRTIM_TIMDEADTIME_RISINGLOCK_READONLY | Deadtime rising value and sign is read-only |

***HRTIM Deadtime Rising Sign***

|                                       |                                  |
|---------------------------------------|----------------------------------|
| HRTIM_TIMDEADTIME_RISINGSIGN_POSITIVE | Positive deadtime on rising edge |
| HRTIM_TIMDEADTIME_RISINGSIGN_NEGATIVE | Negative deadtime on rising edge |

***HRTIM Deadtime Rising Sign Lock***

|                                           |                                   |
|-------------------------------------------|-----------------------------------|
| HRTIM_TIMDEADTIME_RISINGSIGNLOCK_WRITE    | Deadtime rising sign is writeable |
| HRTIM_TIMDEADTIME_RISINGSIGNLOCK_READONLY | Deadtime rising sign is read-only |

***HRTIM DLL Calibration Rate***

|                            |                                                            |
|----------------------------|------------------------------------------------------------|
| HRTIM_SINGLE_CALIBRATION   | Non periodic DLL calibration                               |
| HRTIM_CALIBRATIONRATE_7300 | Periodic DLL calibration: $T = 1048576U * tHRTIM$ (7.3 ms) |
| HRTIM_CALIBRATIONRATE_910  | Periodic DLL calibration: $T = 131072U * tHRTIM$ (910 ms)  |
| HRTIM_CALIBRATIONRATE_114  | Periodic DLL calibration: $T = 16384U * tHRTIM$ (114 ms)   |
| HRTIM_CALIBRATIONRATE_14   | Periodic DLL calibration: $T = 2048U * tHRTIM$ (14 ms)     |

***HRTIM Exported Macros***

|                                      |                                                                                                                                                 |
|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>_HAL_HRTIM_RESET_HANDLE_STATE</u> | <b>Description:</b><br>• Reset HRTIM handle state.<br><b>Parameters:</b><br>• <u>_HANDLE_</u> : HRTIM handle.<br><b>Return value:</b><br>• None |
|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|

`_HAL_HRTIM_ENABLE`**Description:**

- Enables or disables the timer counter(s)

**Parameters:**

- `_HANDLE_`: specifies the HRTIM Handle.
- `_TIMERS_`: timers to enable/disable This parameter can be any combinations of the following values:
  - `HRTIM_TIMERID_MASTER`: Master timer identifier
  - `HRTIM_TIMERID_TIMER_A`: Timer A identifier
  - `HRTIM_TIMERID_TIMER_B`: Timer B identifier
  - `HRTIM_TIMERID_TIMER_C`: Timer C identifier
  - `HRTIM_TIMERID_TIMER_D`: Timer D identifier
  - `HRTIM_TIMERID_TIMER_E`: Timer E identifier

**Return value:**

- None

`HRTIM_TAOEN_MASK`  
`HRTIM_TBOEN_MASK`  
`HRTIM_TCOEN_MASK`  
`HRTIM_TDOEN_MASK`  
`HRTIM_TEOEN_MASK`  
`_HAL_HRTIM_DISABLE`  
`_HAL_HRTIM_ENABLE_IT`

**Description:**

- Enables or disables the specified HRTIM common interrupts.

**Parameters:**

- `_HANDLE_`: specifies the HRTIM Handle.
- `_INTERRUPT_`: specifies the interrupt source to enable or disable. This parameter can be one of the following values:
  - `HRTIM_IT_FLT1`: Fault 1 interrupt enable
  - `HRTIM_IT_FLT2`: Fault 2 interrupt enable
  - `HRTIM_IT_FLT3`: Fault 3 interrupt enable
  - `HRTIM_IT_FLT4`: Fault 4 interrupt enable
  - `HRTIM_IT_FLT5`: Fault 5 interrupt enable
  - `HRTIM_IT_SYSFLT`: System Fault interrupt enable



- HRTIM\_IT\_DLLRDY: DLL ready interrupt enable
- HRTIM\_IT\_BMPER: Burst mode period interrupt enable

**Return value:**

- None

`_HAL_HRTIM_ENABLE_IT`

**Description:**

- Enables or disables the specified HRTIM common interrupts.

**Parameters:**

- `_HANDLE`: specifies the HRTIM Handle.
- `_INTERRUPT`: specifies the interrupt source to enable or disable. This parameter can be one of the following values:
  - HRTIM\_IT\_FLT1: Fault 1 interrupt enable
  - HRTIM\_IT\_FLT2: Fault 2 interrupt enable
  - HRTIM\_IT\_FLT3: Fault 3 interrupt enable
  - HRTIM\_IT\_FLT4: Fault 4 interrupt enable
  - HRTIM\_IT\_FLT5: Fault 5 interrupt enable
  - HRTIM\_IT\_SYSFLT: System Fault interrupt enable
  - HRTIM\_IT\_DLLRDY: DLL ready interrupt enable
  - HRTIM\_IT\_BMPER: Burst mode period interrupt enable

**Return value:**

- None

`_HAL_HRTIM_DISABLE_IT`

`_HAL_HRTIM_DISABLE_IT`

`_HAL_HRTIM_MASTER_ENABLE_IT`

**Description:**

- Enables or disables the specified HRTIM Master timer interrupts.

**Parameters:**

- `_HANDLE`: specifies the HRTIM Handle.
- `_INTERRUPT`: specifies the interrupt source to enable or disable. This parameter can be one of the following values:
  - HRTIM\_MASTER\_IT\_MCMP1: Master compare 1 interrupt enable
  - HRTIM\_MASTER\_IT\_MCMP2: Master compare 2 interrupt enable
  - HRTIM\_MASTER\_IT\_MCMP3: Master

- compare 3 interrupt enable
- HRTIM\_MASTER\_IT\_MCMP4: Master compare 4 interrupt enable
- HRTIM\_MASTER\_IT\_MREP: Master Repetition interrupt enable
- HRTIM\_MASTER\_IT\_SYNC: Synchronization input interrupt enable
- HRTIM\_MASTER\_IT\_UPD: Master update interrupt enable

**Return value:**

- None

`__HAL_HRTIM_MASTER_DISABLE`

`IT`

`__HAL_HRTIM_TIMER_ENABLE_IT`

**Description:**

- Enables or disables the specified HRTIM Timerx interrupts.

**Parameters:**

- `__HANDLE__`: specifies the HRTIM Handle.
- `__TIMER__`: specified the timing unit (Timer A to E)
- `__INTERRUPT__`: specifies the interrupt source to enable or disable. This parameter can be one of the following values:
  - HRTIM\_TIM\_IT\_CMP1: Timer compare 1 interrupt enable
  - HRTIM\_TIM\_IT\_CMP2: Timer compare 2 interrupt enable
  - HRTIM\_TIM\_IT\_CMP3: Timer compare 3 interrupt enable
  - HRTIM\_TIM\_IT\_CMP4: Timer compare 4 interrupt enable
  - HRTIM\_TIM\_IT REP: Timer repetition interrupt enable
  - HRTIM\_TIM\_IT\_UPD: Timer update interrupt enable
  - HRTIM\_TIM\_IT\_CPT1: Timer capture 1 interrupt enable
  - HRTIM\_TIM\_IT\_CPT2: Timer capture 2 interrupt enable
  - HRTIM\_TIM\_IT\_SET1: Timer output 1 set interrupt enable
  - HRTIM\_TIM\_IT\_RST1: Timer output 1 reset interrupt enable
  - HRTIM\_TIM\_IT\_SET2: Timer output 2 set interrupt enable
  - HRTIM\_TIM\_IT\_RST2: Timer output 2 reset interrupt enable
  - HRTIM\_TIM\_IT\_RST: Timer reset interrupt enable

- HRTIM\_TIM\_IT\_DLYPRT: Timer delay protection interrupt enable

**Return value:**

- None

`_HAL_HRTIM_TIMER_DISABLE_IT`  
`_HAL_HRTIM_GET_ITSTATUS`

**Description:**

- Checks if the specified HRTIM common interrupt source is enabled or disabled.

**Parameters:**

- `_HANDLE_`: specifies the HRTIM Handle.
- `_INTERRUPT_`: specifies the interrupt source to check. This parameter can be one of the following values:
  - HRTIM\_IT\_FLT1: Fault 1 interrupt enable
  - HRTIM\_IT\_FLT2: Fault 2 interrupt enable
  - HRTIM\_IT\_FLT3: Fault 3 enable
  - HRTIM\_IT\_FLT4: Fault 4 enable
  - HRTIM\_IT\_FLT5: Fault 5 enable
  - HRTIM\_IT\_SYSFLT: System Fault interrupt enable
  - HRTIM\_IT\_DLLRDY: DLL ready interrupt enable
  - HRTIM\_IT\_BMPER: Burst mode period interrupt enable

**Return value:**

- The: new state of `_INTERRUPT_` (TRUE or FALSE).

`_HAL_HRTIM_MASTER_GET_IT_STATUS`

**Description:**

- Checks if the specified HRTIM Master interrupt source is enabled or disabled.

**Parameters:**

- `_HANDLE_`: specifies the HRTIM Handle.
- `_INTERRUPT_`: specifies the interrupt source to check. This parameter can be one of the following values:
  - HRTIM\_MASTER\_IT\_MCMP1: Master compare 1 interrupt enable
  - HRTIM\_MASTER\_IT\_MCMP2: Master compare 2 interrupt enable
  - HRTIM\_MASTER\_IT\_MCMP3: Master compare 3 interrupt enable
  - HRTIM\_MASTER\_IT\_MCMP4: Master compare 4 interrupt enable
  - HRTIM\_MASTER\_IT\_MREP: Master Repetition interrupt enable

- HRTIM\_MASTER\_IT\_SYNC:  
Synchronization input interrupt enable
- HRTIM\_MASTER\_IT\_MUPD: Master update interrupt enable

**Return value:**

- The new state of \_\_INTERRUPT\_\_ (TRUE or FALSE).

`__HAL_HRTIM_TIMER_GET_IT_STATUS`

**Description:**

- Checks if the specified HRTIM Timerx interrupt source is enabled or disabled.

**Parameters:**

- `__HANDLE__`: specifies the HRTIM Handle.
- `__TIMER__`: specified the timing unit (Timer A to E)
- `__INTERRUPT__`: specifies the interrupt source to check. This parameter can be one of the following values:
  - HRTIM\_MASTER\_IT\_MCMP1: Master compare 1 interrupt enable
  - HRTIM\_MASTER\_IT\_MCMP2: Master compare 2 interrupt enable
  - HRTIM\_MASTER\_IT\_MCMP3: Master compare 3 interrupt enable
  - HRTIM\_MASTER\_IT\_MCMP4: Master compare 4 interrupt enable
  - HRTIM\_MASTER\_IT\_MREP: Master Repetition interrupt enable
  - HRTIM\_MASTER\_IT\_SYNC:  
Synchronization input interrupt enable
  - HRTIM\_MASTER\_IT\_MUPD: Master update interrupt enable
  - HRTIM\_TIM\_IT\_CMP1: Timer compare 1 interrupt enable
  - HRTIM\_TIM\_IT\_CMP2: Timer compare 2 interrupt enable
  - HRTIM\_TIM\_IT\_CMP3: Timer compare 3 interrupt enable
  - HRTIM\_TIM\_IT\_CMP4: Timer compare 4 interrupt enable
  - HRTIM\_TIM\_IT REP: Timer repetition interrupt enable
  - HRTIM\_TIM\_IT\_UPD: Timer update interrupt enable
  - HRTIM\_TIM\_IT\_CPT1: Timer capture 1 interrupt enable
  - HRTIM\_TIM\_IT\_CPT2: Timer capture 2 interrupt enable
  - HRTIM\_TIM\_IT\_SET1: Timer output 1 set interrupt enable
  - HRTIM\_TIM\_IT\_RST1: Timer output 1 reset interrupt enable



- HRTIM\_TIM\_IT\_SET2: Timer output 2 set interrupt enable
- HRTIM\_TIM\_IT\_RST2: Timer output 2 reset interrupt enable
- HRTIM\_TIM\_IT\_RST: Timer reset interrupt enable
- HRTIM\_TIM\_IT\_DLYPRT: Timer delay protection interrupt enable

**Return value:**

- The new state of \_\_INTERRUPT\_\_ (TRUE or FALSE).

[\\_\\_HAL\\_HRTIM\\_CLEAR\\_IT](#)**Description:**

- Clears the specified HRTIM common pending flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the HRTIM Handle.
- \_\_INTERRUPT\_\_: specifies the interrupt pending bit to clear. This parameter can be one of the following values:
  - HRTIM\_IT\_FLT1: Fault 1 interrupt clear flag
  - HRTIM\_IT\_FLT2: Fault 2 interrupt clear flag
  - HRTIM\_IT\_FLT3: Fault 3 clear flag
  - HRTIM\_IT\_FLT4: Fault 4 clear flag
  - HRTIM\_IT\_FLT5: Fault 5 clear flag
  - HRTIM\_IT\_SYSFLT: System Fault interrupt clear flag
  - HRTIM\_IT\_DLLRDY: DLL ready interrupt clear flag
  - HRTIM\_IT\_BMPER: Burst mode period interrupt clear flag

**Return value:**

- None

[\\_\\_HAL\\_HRTIM\\_MASTER\\_CLEAR\\_IT](#)**Description:**

- Clears the specified HRTIM Master pending flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the HRTIM Handle.
- \_\_INTERRUPT\_\_: specifies the interrupt pending bit to clear. This parameter can be one of the following values:
  - HRTIM\_MASTER\_IT\_MCMP1: Master compare 1 interrupt clear flag
  - HRTIM\_MASTER\_IT\_MCMP2: Master compare 2 interrupt clear flag
  - HRTIM\_MASTER\_IT\_MCMP3: Master compare 3 interrupt clear flag

- HRTIM\_MASTER\_IT\_MCMP4: Master compare 4 interrupt clear flag
- HRTIM\_MASTER\_IT\_MREP: Master Repetition interrupt clear flag
- HRTIM\_MASTER\_IT\_SYNC: Synchronization input interrupt clear flag
- HRTIM\_MASTER\_IT\_MUPD: Master update interrupt clear flag

**Return value:**

- None

**\_HAL\_HRTIM\_TIMER\_CLEAR\_IT****Description:**

- Clears the specified HRTIM Timerx pending flag.

**Parameters:**

- HANDLE: specifies the HRTIM Handle.
- TIMER: specified the timing unit (Timer A to E)
- INTERRUPT: specifies the interrupt pending bit to clear. This parameter can be one of the following values:
  - HRTIM\_TIM\_IT\_CMP1: Timer compare 1 interrupt clear flag
  - HRTIM\_TIM\_IT\_CMP2: Timer compare 2 interrupt clear flag
  - HRTIM\_TIM\_IT\_CMP3: Timer compare 3 interrupt clear flag
  - HRTIM\_TIM\_IT\_CMP4: Timer compare 4 interrupt clear flag
  - HRTIM\_TIM\_IT REP: Timer repetition interrupt clear flag
  - HRTIM\_TIM\_IT\_UPD: Timer update interrupt clear flag
  - HRTIM\_TIM\_IT\_CPT1: Timer capture 1 interrupt clear flag
  - HRTIM\_TIM\_IT\_CPT2: Timer capture 2 interrupt clear flag
  - HRTIM\_TIM\_IT\_SET1: Timer output 1 set interrupt clear flag
  - HRTIM\_TIM\_IT\_RST1: Timer output 1 reset interrupt clear flag
  - HRTIM\_TIM\_IT\_SET2: Timer output 2 set interrupt clear flag
  - HRTIM\_TIM\_IT\_RST2: Timer output 2 reset interrupt clear flag
  - HRTIM\_TIM\_IT\_RST: Timer reset interrupt clear flag
  - HRTIM\_TIM\_IT\_DLYPRT: Timer output 1 delay protection interrupt clear flag

**Return value:**

`_HAL_HRTIM_MASTER_ENABLE_  
DMA`

- None

**Description:**

- Enables or disables the specified HRTIM Master timer DMA requests.

**Parameters:**

- `_HANDLE_`: specifies the HRTIM Handle.
- `_DMA_`: specifies the DMA request to enable or disable. This parameter can be one of the following values:
  - `HRTIM_MASTER_DMA_MCMP1`: Master compare 1 DMA request enable
  - `HRTIM_MASTER_DMA_MCMP2`: Master compare 2 DMA request enable
  - `HRTIM_MASTER_DMA_MCMP3`: Master compare 3 DMA request enable
  - `HRTIM_MASTER_DMA_MCMP4`: Master compare 4 DMA request enable
  - `HRTIM_MASTER_DMA_MREP`: Master Repetition DMA request enable
  - `HRTIM_MASTER_DMA_SYNC`: Synchronization input DMA request enable
  - `HRTIM_MASTER_DMA_MUPD`: Master update DMA request enable

**Return value:**

- None

`_HAL_HRTIM_MASTER_DISABLE  
DMA`

`_HAL_HRTIM_TIMER_ENABLE_  
DMA`

**Description:**

- Enables or disables the specified HRTIM Timerx DMA requests.

**Parameters:**

- `_HANDLE_`: specifies the HRTIM Handle.
- `_TIMER_`: specified the timing unit (Timer A to E)
- `_DMA_`: specifies the DMA request to enable or disable. This parameter can be one of the following values:
  - `HRTIM_TIM_DMA_CMP1`: Timer compare 1 DMA request enable
  - `HRTIM_TIM_DMA_CMP2`: Timer compare 2 DMA request enable
  - `HRTIM_TIM_DMA_CMP3`: Timer compare 3 DMA request enable
  - `HRTIM_TIM_DMA_CMP4`: Timer compare 4 DMA request enable
  - `HRTIM_TIM_DMA REP`: Timer

- repetition DMA resquest enable
- HRTIM\_TIM\_DMA\_UPD: Timer update DMA resquest enable
- HRTIM\_TIM\_DMA\_CPT1: Timer capture 1 DMA resquest enable
- HRTIM\_TIM\_DMA\_CPT2: Timer capture 2 DMA resquest enable
- HRTIM\_TIM\_DMA\_SET1: Timer output 1 set DMA resquest enable
- HRTIM\_TIM\_DMA\_RST1: Timer output 1 reset DMA resquest enable
- HRTIM\_TIM\_DMA\_SET2: Timer output 2 set DMA resquest enable
- HRTIM\_TIM\_DMA\_RST2: Timer output 2 reset DMA resquest enable
- HRTIM\_TIM\_DMA\_RST: Timer reset DMA resquest enable
- HRTIM\_TIM\_DMA\_DLYPRT: Timer delay protection DMA resquest enable

**Return value:**

- None

`_HAL_HRTIM_TIMER_DISABLE_  
DMA`  
`_HAL_HRTIM_GET_FLAG`  
`_HAL_HRTIM_CLEAR_FLAG`  
`_HAL_HRTIM_MASTER_GET_  
FLAG`  
`_HAL_HRTIM_MASTER_CLEAR_  
FLAG`  
`_HAL_HRTIM_TIMER_GET_FLAG`  
`_HAL_HRTIM_TIMER_CLEAR_  
FLAG`  
`_HAL_HRTIM_SETCOUNTER`

**Description:**

- Sets the HRTIM timer Counter Register value on runtime.

**Parameters:**

- `_HANDLE_`: HRTIM Handle.
- `_TIMER_`: HRTIM timer This parameter can be one of the following values:
  - 0x5 for master timer
  - 0x0 to 0x4 for timers A to E
- `_COUNTER_`: specifies the Counter Register new value.

**Return value:**

- None

\_\_HAL\_HRTIM\_GETCOUNTER**Description:**

- Gets the HRTIM timer Counter Register value on runtime.

**Parameters:**

- \_\_HANDLE\_\_: HRTIM Handle.
- \_\_TIMER\_\_: HRTIM timer This parameter can be one of the following values:
  - 0x5 for master timer
  - 0x0 to 0x4 for timers A to E

**Return value:**

- HRTIM: timer Counter Register value

\_\_HAL\_HRTIM\_SETPERIOD**Description:**

- Sets the HRTIM timer Period value on runtime.

**Parameters:**

- \_\_HANDLE\_\_: HRTIM Handle.
- \_\_TIMER\_\_: HRTIM timer This parameter can be one of the following values:
  - 0x5 for master timer
  - 0x0 to 0x4 for timers A to E
- \_\_PERIOD\_\_: specifies the Period Register new value.

**Return value:**

- None

\_\_HAL\_HRTIM\_GETPERIOD**Description:**

- Gets the HRTIM timer Period Register value on runtime.

**Parameters:**

- \_\_HANDLE\_\_: HRTIM Handle.
- \_\_TIMER\_\_: HRTIM timer This parameter can be one of the following values:
  - 0x5 for master timer
  - 0x0 to 0x4 for timers A to E

**Return value:**

- timer: Period Register

\_\_HAL\_HRTIM\_SETCLOCK  
PRESCALER**Description:**

- Sets the HRTIM timer clock prescaler value on runtime.

**Parameters:**

- \_\_HANDLE\_\_: HRTIM Handle.
- \_\_TIMER\_\_: HRTIM timer This parameter can be one of the following values:
  - 0x5 for master timer

- 0x0 to 0x4 for timers A to E
- PRESALER: specifies the clock prescaler new value. This parameter can be one of the following values:
  - HRTIM\_PRESCALERRATIO\_MUL32:  
fHRCK: 4.608 GHz - Resolution: 217 ps  
- Min PWM frequency: 70.3 kHz  
(fHRTIM=144MHz)
  - HRTIM\_PRESCALERRATIO\_MUL16:  
fHRCK: 2.304 GHz - Resolution: 434 ps  
- Min PWM frequency: 35.1 KHz  
(fHRTIM=144MHz)
  - HRTIM\_PRESCALERRATIO\_MUL8:  
fHRCK: 1.152 GHz - Resolution: 868 ps  
- Min PWM frequency: 17.6 kHz  
(fHRTIM=144MHz)
  - HRTIM\_PRESCALERRATIO\_MUL4:  
fHRCK: 576 MHz - Resolution: 1.73 ns -  
Min PWM frequency: 8.8 kHz  
(fHRTIM=144MHz)
  - HRTIM\_PRESCALERRATIO\_MUL2:  
fHRCK: 288 MHz - Resolution: 3.47 ns -  
Min PWM frequency: 4.4 kHz  
(fHRTIM=144MHz)
  - HRTIM\_PRESCALERRATIO\_DIV1:  
fHRCK: 144 MHz - Resolution: 6.95 ns -  
Min PWM frequency: 2.2 kHz  
(fHRTIM=144MHz)
  - HRTIM\_PRESCALERRATIO\_DIV2:  
fHRCK: 72 MHz - Resolution: 13.88 ns-  
Min PWM frequency: 1.1 kHz  
(fHRTIM=144MHz)
  - HRTIM\_PRESCALERRATIO\_DIV4:  
fHRCK: 36 MHz - Resolution: 27.7 ns-  
Min PWM frequency: 550Hz  
(fHRTIM=144MHz)

**Return value:**

- None

**Description:**

- Gets the HRTIM timer clock prescaler value on runtime.

**Parameters:**

- HANDLE: HRTIM Handle.
- TIMER: HRTIM timer This parameter can be one of the following values:
  - 0x5 for master timer
  - 0x0 to 0x4 for timers A to E

**Return value:**

- timer: clock prescaler value

[\\_\\_HAL\\_HRTIM\\_SETCOMPARE](#)**Description:**

- Sets the HRTIM timer Compare Register value on runtime.
- Parameters:**
- [\\_\\_HANDLE\\_\\_](#): HRTIM Handle.
  - [\\_\\_TIMER\\_\\_](#): HRTIM timer This parameter can be one of the following values:
    - 0x0 to 0x4 for timers A to E
  - [\\_\\_COMPAREUNIT\\_\\_](#): timer compare unit This parameter can be one of the following values:
    - [HRTIM\\_COMPAREUNIT\\_1](#): Compare unit 1
    - [HRTIM\\_COMPAREUNIT\\_2](#): Compare unit 2
    - [HRTIM\\_COMPAREUNIT\\_3](#): Compare unit 3
    - [HRTIM\\_COMPAREUNIT\\_4](#): Compare unit 4
  - [\\_\\_COMPARE\\_\\_](#): specifies the Compare new value.

**Return value:**

- None

[\\_\\_HAL\\_HRTIM\\_GETCOMPARE](#)**Description:**

- Gets the HRTIM timer Compare Register value on runtime.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): HRTIM Handle.
- [\\_\\_TIMER\\_\\_](#): HRTIM timer This parameter can be one of the following values:
  - 0x0 to 0x4 for timers A to E
- [\\_\\_COMPAREUNIT\\_\\_](#): timer compare unit This parameter can be one of the following values:
  - [HRTIM\\_COMPAREUNIT\\_1](#): Compare unit 1
  - [HRTIM\\_COMPAREUNIT\\_2](#): Compare unit 2
  - [HRTIM\\_COMPAREUNIT\\_3](#): Compare unit 3
  - [HRTIM\\_COMPAREUNIT\\_4](#): Compare unit 4

**Return value:**

- Compare: value

***HRTIM External Event Channels***

|                  |                                      |
|------------------|--------------------------------------|
| HRTIM_EVENT_NONE | Undefined event channel              |
| HRTIM_EVENT_1    | External event channel 1 identifier  |
| HRTIM_EVENT_2    | External event channel 2 identifier  |
| HRTIM_EVENT_3    | External event channel 3 identifier  |
| HRTIM_EVENT_4    | External event channel 4 identifier  |
| HRTIM_EVENT_5    | External event channel 5 identifier  |
| HRTIM_EVENT_6    | External event channel 6 identifier  |
| HRTIM_EVENT_7    | External event channel 7 identifier  |
| HRTIM_EVENT_8    | External event channel 8 identifier  |
| HRTIM_EVENT_9    | External event channel 9 identifier  |
| HRTIM_EVENT_10   | External event channel 10 identifier |

***HRTIM External Event Fast Mode***

|                             |                                                                               |
|-----------------------------|-------------------------------------------------------------------------------|
| HRTIM_EVENTFASTMODE_ENABLE  | External Event is re-synchronized by the HRTIM logic before acting on outputs |
| HRTIM_EVENTFASTMODE_DISABLE | External Event is acting asynchronously on outputs (low latency mode)         |

***HRTIM External Event Filter***

|                        |                            |
|------------------------|----------------------------|
| HRTIM_EVENTFILTER_NONE | Filter disabled            |
| HRTIM_EVENTFILTER_1    | fSAMPLING= fHRTIM, N=2U    |
| HRTIM_EVENTFILTER_2    | fSAMPLING= fHRTIM, N=4U    |
| HRTIM_EVENTFILTER_3    | fSAMPLING= fHRTIM, N=8U    |
| HRTIM_EVENTFILTER_4    | fSAMPLING= fEEVS/2U, N=6U  |
| HRTIM_EVENTFILTER_5    | fSAMPLING= fEEVS/2U, N=8U  |
| HRTIM_EVENTFILTER_6    | fSAMPLING= fEEVS/4U, N=6U  |
| HRTIM_EVENTFILTER_7    | fSAMPLING= fEEVS/4U, N=8U  |
| HRTIM_EVENTFILTER_8    | fSAMPLING= fEEVS/8U, N=6U  |
| HRTIM_EVENTFILTER_9    | fSAMPLING= fEEVS/8U, N=8U  |
| HRTIM_EVENTFILTER_10   | fSAMPLING= fEEVS/16U, N=5U |
| HRTIM_EVENTFILTER_11   | fSAMPLING= fEEVS/16U, N=6U |
| HRTIM_EVENTFILTER_12   | fSAMPLING= fEEVS/16U, N=8U |
| HRTIM_EVENTFILTER_13   | fSAMPLING= fEEVS/32U, N=5U |
| HRTIM_EVENTFILTER_14   | fSAMPLING= fEEVS/32U, N=6U |
| HRTIM_EVENTFILTER_15   | fSAMPLING= fEEVS/32U, N=8U |

***HRTIM External Event Polarity***

|                          |                               |
|--------------------------|-------------------------------|
| HRTIM_EVENTPOLARITY_HIGH | External event is active high |
| HRTIM_EVENTPOLARITY_LOW  | External event is active low  |

***HRTIM External Event Prescaler***

|                           |                   |
|---------------------------|-------------------|
| HRTIM_EVENTPRESCALER_DIV1 | fEEVS=fHRTIM      |
| HRTIM_EVENTPRESCALER_DIV2 | fEEVS=fHRTIM / 2U |
| HRTIM_EVENTPRESCALER_DIV4 | fEEVS=fHRTIM / 4U |
| HRTIM_EVENTPRESCALER_DIV8 | fEEVS=fHRTIM / 8U |

***HRTIM External Event Sensitivity***

|                                    |                                                      |
|------------------------------------|------------------------------------------------------|
| HRTIM_EVENTSensitivity_LEVEL       | External event is active on level                    |
| HRTIM_EVENTSensitivity_RISINGEDGE  | External event is active on Rising edge              |
| HRTIM_EVENTSensitivity_FALLINGEDGE | External event is active on Falling edge             |
| HRTIM_EVENTSensitivity_BOTHEDGES   | External event is active on Rising and Falling edges |

***HRTIM External Event Sources***

|                  |                          |
|------------------|--------------------------|
| HRTIM_EVENTSRC_1 | External event source 1U |
| HRTIM_EVENTSRC_2 | External event source 2U |
| HRTIM_EVENTSRC_3 | External event source 3U |
| HRTIM_EVENTSRC_4 | External event source 4U |

***HRTIM External Fault Prescaler***

|                           |                   |
|---------------------------|-------------------|
| HRTIM_FAULTPRESCALER_DIV1 | fFLTS=fHRTIM      |
| HRTIM_FAULTPRESCALER_DIV2 | fFLTS=fHRTIM / 2U |
| HRTIM_FAULTPRESCALER_DIV4 | fFLTS=fHRTIM / 4U |
| HRTIM_FAULTPRESCALER_DIV8 | fFLTS=fHRTIM / 8U |

***HRTIM Fault Channel***

|               |                            |
|---------------|----------------------------|
| HRTIM_FAULT_1 | Fault channel 1 identifier |
| HRTIM_FAULT_2 | Fault channel 2 identifier |
| HRTIM_FAULT_3 | Fault channel 3 identifier |
| HRTIM_FAULT_4 | Fault channel 4 identifier |
| HRTIM_FAULT_5 | Fault channel 5 identifier |

***HRTIM Fault Filter***

|                        |                           |
|------------------------|---------------------------|
| HRTIM_FAULTFILTER_NONE | Filter disabled           |
| HRTIM_FAULTFILTER_1    | fSAMPLING= fHRTIM, N=2U   |
| HRTIM_FAULTFILTER_2    | fSAMPLING= fHRTIM, N=4U   |
| HRTIM_FAULTFILTER_3    | fSAMPLING= fHRTIM, N=8U   |
| HRTIM_FAULTFILTER_4    | fSAMPLING= fFLTS/2U, N=6U |
| HRTIM_FAULTFILTER_5    | fSAMPLING= fFLTS/2U, N=8U |
| HRTIM_FAULTFILTER_6    | fSAMPLING= fFLTS/4U, N=6U |
| HRTIM_FAULTFILTER_7    | fSAMPLING= fFLTS/4U, N=8U |
| HRTIM_FAULTFILTER_8    | fSAMPLING= fFLTS/8U, N=6U |

|                                               |                                                                               |
|-----------------------------------------------|-------------------------------------------------------------------------------|
| HRTIM_FAULTFILTER_9                           | fSAMPLING= fFLTS/8U, N=8U                                                     |
| HRTIM_FAULTFILTER_10                          | fSAMPLING= fFLTS/16U, N=5U                                                    |
| HRTIM_FAULTFILTER_11                          | fSAMPLING= fFLTS/16U, N=6U                                                    |
| HRTIM_FAULTFILTER_12                          | fSAMPLING= fFLTS/16U, N=8U                                                    |
| HRTIM_FAULTFILTER_13                          | fSAMPLING= fFLTS/32U, N=5U                                                    |
| HRTIM_FAULTFILTER_14                          | fSAMPLING= fFLTS/32U, N=6U                                                    |
| HRTIM_FAULTFILTER_15                          | fSAMPLING= fFLTS/32U, N=8U                                                    |
| <b><i>HRTIM Fault Lock</i></b>                |                                                                               |
| HRTIM_FAULTLOCK_READWRITE                     | Fault settings bits are read/write                                            |
| HRTIM_FAULTLOCK_READONLY                      | Fault settings bits are read only                                             |
| <b><i>HRTIM Fault Mode Control</i></b>        |                                                                               |
| HRTIM_FAULTMODECTL_DISABLED                   | Fault channel is disabled                                                     |
| HRTIM_FAULTMODECTL_ENABLED                    | Fault channel is enabled                                                      |
| IS_HRTIM_FAULTMODECTL                         |                                                                               |
| <b><i>HRTIM Fault Polarity</i></b>            |                                                                               |
| HRTIM_FAULTPOLARITY_LOW                       | Fault input is active low                                                     |
| HRTIM_FAULTPOLARITY_HIGH                      | Fault input is active high                                                    |
| <b><i>HRTIM Fault Sources</i></b>             |                                                                               |
| HRTIM_FAULTSOURCE_DIGITALINPUT                | Fault input is FLT input pin                                                  |
| HRTIM_FAULTSOURCE_INTERNAL                    | Fault input is FLT_Int signal (e.g. internal comparator)                      |
| <b><i>HRTIM Half Mode Enable</i></b>          |                                                                               |
| HRTIM_HALFMODE_DISABLED                       | Half mode is disabled                                                         |
| HRTIM_HALFMODE_ENABLED                        | Half mode is enabled                                                          |
| <b><i>HRTIM Idle Push Pull Status</i></b>     |                                                                               |
| HRTIM_PUSHPULL_IDLESTATUS_OUTPUT1             | Protection occurred when the output 1 was active and output 2 forced inactive |
| HRTIM_PUSHPULL_IDLESTATUS_OUTPUT2             | Protection occurred when the output 2 was active and output 1 forced inactive |
| <b><i>HRTIM Master DMA Request Enable</i></b> |                                                                               |
| HRTIM_MASTER_DMA_NONE                         | No DMA request enable                                                         |
| HRTIM_MASTER_DMA_MCMP1                        | Master compare 1 DMA request enable                                           |
| HRTIM_MASTER_DMA_MCMP2                        | Master compare 2 DMA request enable                                           |
| HRTIM_MASTER_DMA_MCMP3                        | Master compare 3 DMA request enable                                           |
| HRTIM_MASTER_DMA_MCMP4                        | Master compare 4 DMA request enable                                           |
| HRTIM_MASTER_DMA_MREP                         | Master Repetition DMA request enable                                          |
| HRTIM_MASTER_DMA_SYNC                         | Synchronization input DMA request enable                                      |

**HRTIM\_MASTER\_DMA\_MUPD** Master update DMA request enable

***HRTIM Master Interrupt Enable***

**HRTIM\_MASTER\_IT\_NONE** No interrupt enabled

**HRTIM\_MASTER\_IT\_MCMP1** Master compare 1 interrupt enable

**HRTIM\_MASTER\_IT\_MCMP2** Master compare 2 interrupt enable

**HRTIM\_MASTER\_IT\_MCMP3** Master compare 3 interrupt enable

**HRTIM\_MASTER\_IT\_MCMP4** Master compare 4 interrupt enable

**HRTIM\_MASTER\_IT\_MREP** Master Repetition interrupt enable

**HRTIM\_MASTER\_IT\_SYNC** Synchronization input interrupt enable

**HRTIM\_MASTER\_IT\_MUPD** Master update interrupt enable

***HRTIM Master Interrupt Flag***

**HRTIM\_MASTER\_FLAG\_MCMP1** Master compare 1 interrupt flag

**HRTIM\_MASTER\_FLAG\_MCMP2** Master compare 2 interrupt flag

**HRTIM\_MASTER\_FLAG\_MCMP3** Master compare 3 interrupt flag

**HRTIM\_MASTER\_FLAG\_MCMP4** Master compare 4 interrupt flag

**HRTIM\_MASTER\_FLAG\_MREP** Master Repetition interrupt flag

**HRTIM\_MASTER\_FLAG\_SYNC** Synchronization input interrupt flag

**HRTIM\_MASTER\_FLAG\_MUPD** Master update interrupt flag

***HRTIM Max Timer***

**MAX\_HRTIM\_TIMER**

***HRTIM Output Burst Mode Entry Delayed***

**HRTIM\_OUTPUTBURSTMODEENTRY\_REGULAR** The programmed Idle state is applied immediately to the Output

**HRTIM\_OUTPUTBURSTMODEENTRY\_DELAYED** Deadtime is inserted on output before entering the idle mode

***HRTIM Output Chopper Mode Enable***

**HRTIM\_OUTPUTCHOPPERMODE\_DISABLED** Output signal is not altered

**HRTIM\_OUTPUTCHOPPERMODE\_ENABLED** Output signal is chopped by a carrier signal

***HRTIM Output FAULT Level***

**HRTIM\_OUTPUTFAULTLEVEL\_NONE** The output is not affected by the fault input

**HRTIM\_OUTPUTFAULTLEVEL\_ACTIVE** Output at active level when in FAULT state

**HRTIM\_OUTPUTFAULTLEVEL\_INACTIVE** Output at inactive level when in FAULT state

**HRTIM\_OUTPUTFAULTLEVEL\_HIGHZ** Output is tri-stated when in FAULT state

***HRTIM Output IDLE Level***

**HRTIM\_OUTPUTIDLELEVEL\_INACTIVE** Output at inactive level when in IDLE state

**HRTIM\_OUTPUTIDLELEVEL\_ACTIVE** Output at active level when in IDLE state

***HRTIM Output Idle Mode***

|                           |                                                                         |
|---------------------------|-------------------------------------------------------------------------|
| HRTIM_OUTPUTIDLEMODE_NONE | The output is not affected by the burst mode operation                  |
| HRTIM_OUTPUTIDLEMODE_IDLE | The output is in idle state when requested by the burst mode controller |

***HRTIM Output Level***

|                             |                                         |
|-----------------------------|-----------------------------------------|
| HRTIM_OUTPUTLEVEL_ACTIVE    | Forces the output to its active state   |
| HRTIM_OUTPUTLEVEL_INACTIVE  | Forces the output to its inactive state |
| <b>IS_HRTIM_OUTPUTLEVEL</b> |                                         |

***HRTIM Output Polarity***

|                           |                       |
|---------------------------|-----------------------|
| HRTIM_OUTPUTPOLARITY_HIGH | Output is active HIGH |
| HRTIM_OUTPUTPOLARITY_LOW  | Output is active LOW  |

***HRTIM Output Reset Source***

|                              |                                                                                                     |
|------------------------------|-----------------------------------------------------------------------------------------------------|
| HRTIM_OUTPUTRESET_NONE       | Reset the output reset crossbar                                                                     |
| HRTIM_OUTPUTRESET_RESYNC     | Timer reset event coming solely from software or SYNC input forces the output to its inactive state |
| HRTIM_OUTPUTRESET_TIMPER     | Timer period event forces the output to its inactive state                                          |
| HRTIM_OUTPUTRESET_TIMCMP1    | Timer compare 1 event forces the output to its inactive state                                       |
| HRTIM_OUTPUTRESET_TIMCMP2    | Timer compare 2 event forces the output to its inactive state                                       |
| HRTIM_OUTPUTRESET_TIMCMP3    | Timer compare 3 event forces the output to its inactive state                                       |
| HRTIM_OUTPUTRESET_TIMCMP4    | Timer compare 4 event forces the output to its inactive state                                       |
| HRTIM_OUTPUTRESET_MASTERPER  | The master timer period event forces the output to its inactive state                               |
| HRTIM_OUTPUTRESET_MASTERCMP1 | Master Timer compare 1 event forces the output to its inactive state                                |
| HRTIM_OUTPUTRESET_MASTERCMP2 | Master Timer compare 2 event forces the output to its inactive state                                |
| HRTIM_OUTPUTRESET_MASTERCMP3 | Master Timer compare 3 event forces the output to its inactive state                                |
| HRTIM_OUTPUTRESET_MASTERCMP4 | Master Timer compare 4 event forces the output to its inactive state                                |
| HRTIM_OUTPUTRESET_TIMEV_1    | Timer event 1 forces the output to its inactive state                                               |
| HRTIM_OUTPUTRESET_TIMEV_2    | Timer event 2 forces the output to its inactive state                                               |

|                                |                                                                                                   |
|--------------------------------|---------------------------------------------------------------------------------------------------|
| HRTIM_OUTPUTRESET_TIMEV_3      | Timer event 3 forces the output to its inactive state                                             |
| HRTIM_OUTPUTRESET_TIMEV_4      | Timer event 4 forces the output to its inactive state                                             |
| HRTIM_OUTPUTRESET_TIMEV_5      | Timer event 5 forces the output to its inactive state                                             |
| HRTIM_OUTPUTRESET_TIMEV_6      | Timer event 6 forces the output to its inactive state                                             |
| HRTIM_OUTPUTRESET_TIMEV_7      | Timer event 7 forces the output to its inactive state                                             |
| HRTIM_OUTPUTRESET_TIMEV_8      | Timer event 8 forces the output to its inactive state                                             |
| HRTIM_OUTPUTRESET_TIMEV_9      | Timer event 9 forces the output to its inactive state                                             |
| HRTIM_OUTPUTRESET_EEV_1        | External event 1 forces the output to its inactive state                                          |
| HRTIM_OUTPUTRESET_EEV_2        | External event 2 forces the output to its inactive state                                          |
| HRTIM_OUTPUTRESET_EEV_3        | External event 3 forces the output to its inactive state                                          |
| HRTIM_OUTPUTRESET_EEV_4        | External event 4 forces the output to its inactive state                                          |
| HRTIM_OUTPUTRESET_EEV_5        | External event 5 forces the output to its inactive state                                          |
| HRTIM_OUTPUTRESET_EEV_6        | External event 6 forces the output to its inactive state                                          |
| HRTIM_OUTPUTRESET_EEV_7        | External event 7 forces the output to its inactive state                                          |
| HRTIM_OUTPUTRESET_EEV_8        | External event 8 forces the output to its inactive state                                          |
| HRTIM_OUTPUTRESET_EEV_9        | External event 9 forces the output to its inactive state                                          |
| HRTIM_OUTPUTRESET_EEV_10       | External event 10 forces the output to its inactive state                                         |
| HRTIM_OUTPUTRESET_UPDATE       | Timer register update event forces the output to its inactive state                               |
| <b>HRTIM Output Set Source</b> |                                                                                                   |
| HRTIM_OUTPUTSET_NONE           | Reset the output set crossbar                                                                     |
| HRTIM_OUTPUTSET_RESYNC         | Timer reset event coming solely from software or SYNC input forces the output to its active state |
| HRTIM_OUTPUTSET_TIMPER         | Timer period event forces the output to its active state                                          |
| HRTIM_OUTPUTSET_TIMCMP1        | Timer compare 1 event forces the output to its active state                                       |

|                            |                                                                     |
|----------------------------|---------------------------------------------------------------------|
| HRTIM_OUTPUTSET_TIMCMP2    | Timer compare 2 event forces the output to its active state         |
| HRTIM_OUTPUTSET_TIMCMP3    | Timer compare 3 event forces the output to its active state         |
| HRTIM_OUTPUTSET_TIMCMP4    | Timer compare 4 event forces the output to its active state         |
| HRTIM_OUTPUTSET_MASTERPER  | The master timer period event forces the output to its active state |
| HRTIM_OUTPUTSET_MASTERCMP1 | Master Timer compare 1 event forces the output to its active state  |
| HRTIM_OUTPUTSET_MASTERCMP2 | Master Timer compare 2 event forces the output to its active state  |
| HRTIM_OUTPUTSET_MASTERCMP3 | Master Timer compare 3 event forces the output to its active state  |
| HRTIM_OUTPUTSET_MASTERCMP4 | Master Timer compare 4 event forces the output to its active state  |
| HRTIM_OUTPUTSET_TIMEV_1    | Timer event 1 forces the output to its active state                 |
| HRTIM_OUTPUTSET_TIMEV_2    | Timer event 2 forces the output to its active state                 |
| HRTIM_OUTPUTSET_TIMEV_3    | Timer event 3 forces the output to its active state                 |
| HRTIM_OUTPUTSET_TIMEV_4    | Timer event 4 forces the output to its active state                 |
| HRTIM_OUTPUTSET_TIMEV_5    | Timer event 5 forces the output to its active state                 |
| HRTIM_OUTPUTSET_TIMEV_6    | Timer event 6 forces the output to its active state                 |
| HRTIM_OUTPUTSET_TIMEV_7    | Timer event 7 forces the output to its active state                 |
| HRTIM_OUTPUTSET_TIMEV_8    | Timer event 8 forces the output to its active state                 |
| HRTIM_OUTPUTSET_TIMEV_9    | Timer event 9 forces the output to its active state                 |
| HRTIM_OUTPUTSET_EEV_1      | External event 1 forces the output to its active state              |
| HRTIM_OUTPUTSET_EEV_2      | External event 2 forces the output to its active state              |
| HRTIM_OUTPUTSET_EEV_3      | External event 3 forces the output to its active state              |
| HRTIM_OUTPUTSET_EEV_4      | External event 4 forces the output to its active state              |
| HRTIM_OUTPUTSET_EEV_5      | External event 5 forces the output to its active state              |
| HRTIM_OUTPUTSET_EEV_6      | External event 6 forces the output to its active state              |
| HRTIM_OUTPUTSET_EEV_7      | External event 7 forces the output to its active state              |
| HRTIM_OUTPUTSET_EEV_8      | External event 8 forces the output to its active state              |
| HRTIM_OUTPUTSET_EEV_9      | External event 9 forces the output to its active state              |

|                                               |                                                                                                                                 |
|-----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------|
|                                               | state                                                                                                                           |
| HRTIM_OUTPUTSET_EEV_10                        | External event 10 forces the output to its active state                                                                         |
| HRTIM_OUTPUTSET_UPDATE                        | Timer register update event forces the output to its active state                                                               |
| <b><i>HRTIM Output State</i></b>              |                                                                                                                                 |
| HRTIM_OUTPUTSTATE_IDLE                        | Main operating mode, where the output can take the active or inactive level as programmed in the crossbar unit                  |
| HRTIM_OUTPUTSTATE_RUN                         | Default operating state (e.g. after an HRTIM reset, when the outputs are disabled by software or during a burst mode operation) |
| HRTIM_OUTPUTSTATE_FAULT                       | Safety state, entered in case of a shut-down request on FAULTx inputs                                                           |
| <b><i>HRTIM Prescaler Ratio</i></b>           |                                                                                                                                 |
| HRTIM_PRESCALERRATIO_MUL32                    | fHRCK: fHRTIM x 32U = 4.608 GHz - Resolution: 217 ps - Min PWM frequency: 70.3 kHz (fHRTIM=144MHz)                              |
| HRTIM_PRESCALERRATIO_MUL16                    | fHRCK: fHRTIM x 16U = 2.304 GHz - Resolution: 434 ps - Min PWM frequency: 35.1 KHz (fHRTIM=144MHz)                              |
| HRTIM_PRESCALERRATIO_MUL8                     | fHRCK: fHRTIM x 8U = 1.152 GHz - Resolution: 868 ps - Min PWM frequency: 17.6 kHz (fHRTIM=144MHz)                               |
| HRTIM_PRESCALERRATIO_MUL4                     | fHRCK: fHRTIM x 4U = 576 MHz - Resolution: 1.73 ns - Min PWM frequency: 8.8 kHz (fHRTIM=144MHz)                                 |
| HRTIM_PRESCALERRATIO_MUL2                     | fHRCK: fHRTIM x 2U = 288 MHz - Resolution: 3.47 ns - Min PWM frequency: 4.4 kHz (fHRTIM=144MHz)                                 |
| HRTIM_PRESCALERRATIO_DIV1                     | fHRCK: fHRTIM = 144 MHz - Resolution: 6.95 ns - Min PWM frequency: 2.2 kHz (fHRTIM=144MHz)                                      |
| HRTIM_PRESCALERRATIO_DIV2                     | fHRCK: fHRTIM / 2U = 72 MHz - Resolution: 13.88 ns - Min PWM frequency: 1.1 kHz (fHRTIM=144MHz)                                 |
| HRTIM_PRESCALERRATIO_DIV4                     | fHRCK: fHRTIM / 4U = 36 MHz - Resolution: 27.7 ns - Min PWM frequency: 550Hz (fHRTIM=144MHz)                                    |
| <b><i>HRTIM Register Preload Enable</i></b>   |                                                                                                                                 |
| HRTIM_PRELOAD_DISABLED                        | Preload disabled: the write access is directly done into the active register                                                    |
| HRTIM_PRELOAD_ENABLED                         | Preload enabled: the write access is done into the preload register                                                             |
| <b><i>HRTIM Reset On Sync Input Event</i></b> |                                                                                                                                 |
| HRTIM_SYNCRESET_DISABLED                      | Synchronization input event has effect on the timer                                                                             |

|                                                  |                                                                                          |
|--------------------------------------------------|------------------------------------------------------------------------------------------|
| <code>HRTIM_SYNCRESET_ENABLED</code>             | Synchronization input event resets the timer                                             |
| <b><i>HRTIM Simple OC Mode</i></b>               |                                                                                          |
| <code>HRTIM_BASICOCMODE_TOGGLE</code>            | Output toggles when the timer counter reaches the compare value                          |
| <code>HRTIM_BASICOCMODE_INACTIVE</code>          | Output forced to active level when the timer counter reaches the compare value           |
| <code>HRTIM_BASICOCMODE_ACTIVE</code>            | Output forced to inactive level when the timer counter reaches the compare value         |
| <code>IS_HRTIM_BASICOCMODE</code>                |                                                                                          |
| <b><i>HRTIM Software Timer Reset</i></b>         |                                                                                          |
| <code>HRTIM_TIMERRESET_MASTER</code>             | Resets the master timer counter                                                          |
| <code>HRTIM_TIMERRESET_TIMER_A</code>            | Resets the timer A counter                                                               |
| <code>HRTIM_TIMERRESET_TIMER_B</code>            | Resets the timer B counter                                                               |
| <code>HRTIM_TIMERRESET_TIMER_C</code>            | Resets the timer C counter                                                               |
| <code>HRTIM_TIMERRESET_TIMER_D</code>            | Resets the timer D counter                                                               |
| <code>HRTIM_TIMERRESET_TIMER_E</code>            | Resets the timer E counter                                                               |
| <b><i>HRTIM Software Timer Update</i></b>        |                                                                                          |
| <code>HRTIM_TIMERUPDATE_MASTER</code>            | Forces an immediate transfer from the preload to the active register in the master timer |
| <code>HRTIM_TIMERUPDATE_A</code>                 | Forces an immediate transfer from the preload to the active register in the timer A      |
| <code>HRTIM_TIMERUPDATE_B</code>                 | Forces an immediate transfer from the preload to the active register in the timer B      |
| <code>HRTIM_TIMERUPDATE_C</code>                 | Forces an immediate transfer from the preload to the active register in the timer C      |
| <code>HRTIM_TIMERUPDATE_D</code>                 | Forces an immediate transfer from the preload to the active register in the timer D      |
| <code>HRTIM_TIMERUPDATE_E</code>                 | Forces an immediate transfer from the preload to the active register in the timer E      |
| <b><i>HRTIM Start On Sync Input Event</i></b>    |                                                                                          |
| <code>HRTIM_SYNCSTART_DISABLED</code>            | Synchronization input event has effect on the timer                                      |
| <code>HRTIM_SYNCSTART_ENABLED</code>             | Synchronization input event starts the timer                                             |
| <b><i>HRTIM Synchronization Input Source</i></b> |                                                                                          |
| <code>HRTIM_SYNCINPUTSOURCE_NONE</code>          | disabled. HRTIM is not synchronized and runs in standalone mode                          |
| <code>HRTIM_SYNCINPUTSOURCE_INTERNALEVENT</code> | The HRTIM is synchronized with the on-chip timer                                         |
| <code>HRTIM_SYNCINPUTSOURCE_EXTERNALEVENT</code> | A positive pulse on SYNCIN input triggers the HRTIM                                      |

***HRTIM Synchronization Options***

|                         |                                                                                           |
|-------------------------|-------------------------------------------------------------------------------------------|
| HRTIM_SYNCOPTION_NONE   | HRTIM instance doesn't handle external synchronization signals (SYNCIN, SYNCOUT)          |
| HRTIM_SYNCOPTION_MASTER | HRTIM instance acts as a MASTER, i.e. generates external synchronization output (SYNCOUT) |
| HRTIM_SYNCOPTION_SLAVE  | HRTIM instance acts as a SLAVE, i.e. it is synchronized by external sources (SYNCIN)      |

***HRTIM Synchronization Output Polarity***

|                                   |                                                                                                                      |
|-----------------------------------|----------------------------------------------------------------------------------------------------------------------|
| HRTIM_SYNCOUTPUTPOLARITY_NONE     | Synchronization output event is disabled                                                                             |
| HRTIM_SYNCOUTPUTPOLARITY_POSITIVE | SCOUT pin has a low idle level and issues a positive pulse of 16 fHRTIM clock cycles length for the synchronization  |
| HRTIM_SYNCOUTPUTPOLARITY_NEGATIVE | SCOUT pin has a high idle level and issues a negative pulse of 16 fHRTIM clock cycles length for the synchronization |

***HRTIM Synchronization Output Source***

|                                     |                                                                          |
|-------------------------------------|--------------------------------------------------------------------------|
| HRTIM_SYNCOUTPUTSOURCE_MASTER_START | A pulse is sent on the SYNCOUT output upon master timer start event      |
| HRTIM_SYNCOUTPUTSOURCE_MASTER_CMP1  | A pulse is sent on the SYNCOUT output upon master timer compare 1 event  |
| HRTIM_SYNCOUTPUTSOURCE_TIMA_START   | A pulse is sent on the SYNCOUT output upon timer A start or reset events |
| HRTIM_SYNCOUTPUTSOURCE_TIMA_CMP1    | A pulse is sent on the SYNCOUT output upon timer A compare 1 event       |

***HRTIM Timer Burst Mode***

|                                    |                                                                   |
|------------------------------------|-------------------------------------------------------------------|
| HRTIM_TIMERBURSTMODE_MAINTAINCLOCK | Timer counter clock is maintained and the timer operates normally |
| HRTIM_TIMERBURSTMODE_RESETCOUNTER  | Timer counter clock is stopped and the counter is reset           |

***HRTIM Timer Deadtime Insertion***

|                                     |                                                     |
|-------------------------------------|-----------------------------------------------------|
| HRTIM_TIMDEADTIMEINSERTION_DISABLED | Output 1 and output 2 signals are independent       |
| HRTIM_TIMDEADTIMEINSERTION_ENABLED  | Deadtime is inserted between output 1 and output 2U |

***HRTIM Timer Delayed Protection Mode***

|                                             |           |
|---------------------------------------------|-----------|
| HRTIM_TIMER_A_B_C_DELAYEDPROTECTION_DISABLE | No action |
|---------------------------------------------|-----------|

|                                                       |                                                                         |
|-------------------------------------------------------|-------------------------------------------------------------------------|
| HRTIM_TIMER_A_B_C_DELAYEDPROTECTION_DELAYEDOUT1_EEV6  | Timers A, B, C: Output 1 delayed Idle on external Event 6U              |
| HRTIM_TIMER_A_B_C_DELAYEDPROTECTION_DELAYEDOUT2_EEV6  | Timers A, B, C: Output 2 delayed Idle on external Event 6U              |
| HRTIM_TIMER_A_B_C_DELAYEDPROTECTION_DELAYEDBOTH_EEV6  | Timers A, B, C: Output 1 and output 2 delayed Idle on external Event 6U |
| HRTIM_TIMER_A_B_C_DELAYEDPROTECTION_BALANCED_EEV6     | Timers A, B, C: Balanced Idle on external Event 6U                      |
| HRTIM_TIMER_A_B_C_DELAYEDPROTECTION_DELAYEDOUT1_DEEV7 | Timers A, B, C: Output 1 delayed Idle on external Event 7U              |
| HRTIM_TIMER_A_B_C_DELAYEDPROTECTION_DELAYEDOUT2_DEEV7 | Timers A, B, C: Output 2 delayed Idle on external Event 7U              |
| HRTIM_TIMER_A_B_C_DELAYEDPROTECTION_DELAYEDBOTH_EEV7  | Timers A, B, C: Output 1 and output2 delayed Idle on external Event 7U  |
| HRTIM_TIMER_A_B_C_DELAYEDPROTECTION_BALANCED_EEV7     | Timers A, B, C: Balanced Idle on external Event 7U                      |
| HRTIM_TIMER_D_E_DELAYEDPROTECTION_DISABLED            | No action                                                               |
| HRTIM_TIMER_D_E_DELAYEDPROTECTION_DELAYEDOUT1_EEV8    | Timers D, E: Output 1 delayed Idle on external Event 6U                 |
| HRTIM_TIMER_D_E_DELAYEDPROTECTION_DELAYEDOUT2_EEV8    | Timers D, E: Output 2 delayed Idle on external Event 6U                 |
| HRTIM_TIMER_D_E_DELAYEDPROTECTION_DELAYEDBOTH_EEV8    | Timers D, E: Output 1 and output 2 delayed Idle on external Event 6U    |
| HRTIM_TIMER_D_E_DELAYEDPROTECTION_BALANCED_EEV8       | Timers D, E: Balanced Idle on external Event 6U                         |
| HRTIM_TIMER_D_E_DELAYEDPROTECTION_DELAYEDOUT1_DEEV9   | Timers D, E: Output 1 delayed Idle on external Event 7U                 |
| HRTIM_TIMER_D_E_DELAYEDPROTECTION_DELAYEDOUT2_DEEV9   | Timers D, E: Output 2 delayed Idle on external Event 7U                 |
| HRTIM_TIMER_D_E_DELAYEDPROTECTION_DELAYEDBOTH_EEV9    | Timers D, E: Output 1 and output2 delayed Idle on external Event 7U     |
| HRTIM_TIMER_D_E_DELAYEDPROTECTION_BALANCED_EEV9       | Timers D, E: Balanced Idle on external Event 7U                         |
| <b>HRTIM Timer External Event Filter</b>              |                                                                         |
| HRTIM_TIMEVENTFILTER_NONE                             |                                                                         |
| HRTIM_TIMEVENTFILTER_BLANKINGCMP1                     | Blanking from counter reset/roll-over to Compare 1U                     |
| HRTIM_TIMEVENTFILTER_BLANKINGCMP2                     | Blanking from counter reset/roll-over to Compare 2U                     |
| HRTIM_TIMEVENTFILTER_BLANKINGCMP3                     | Blanking from counter reset/roll-over to Compare 3U                     |

|                                    |                                                      |
|------------------------------------|------------------------------------------------------|
| HRTIM_TIMEVENTFILTER_BLANKINGCMP4  | Blanking from counter reset/roll-over to Compare 4U  |
| HRTIM_TIMEVENTFILTER_BLANKINGFLTR1 | Blanking from another timing unit: TIMFLTR1 source   |
| HRTIM_TIMEVENTFILTER_BLANKINGFLTR2 | Blanking from another timing unit: TIMFLTR2 source   |
| HRTIM_TIMEVENTFILTER_BLANKINGFLTR3 | Blanking from another timing unit: TIMFLTR3 source   |
| HRTIM_TIMEVENTFILTER_BLANKINGFLTR4 | Blanking from another timing unit: TIMFLTR4 source   |
| HRTIM_TIMEVENTFILTER_BLANKINGFLTR5 | Blanking from another timing unit: TIMFLTR5 source   |
| HRTIM_TIMEVENTFILTER_BLANKINGFLTR6 | Blanking from another timing unit: TIMFLTR6 source   |
| HRTIM_TIMEVENTFILTER_BLANKINGFLTR7 | Blanking from another timing unit: TIMFLTR7 source   |
| HRTIM_TIMEVENTFILTER_BLANKINGFLTR8 | Blanking from another timing unit: TIMFLTR8 source   |
| HRTIM_TIMEVENTFILTER_WINDOWINGCMP2 | Windowing from counter reset/roll-over to Compare 2U |
| HRTIM_TIMEVENTFILTER_WINDOWINGCMP3 | Windowing from counter reset/roll-over to Compare 3U |
| HRTIM_TIMEVENTFILTER_WINDOWINGTIM  | Windowing from another timing unit: TIMWIN source    |

#### ***HRTIM Timer External Event Latch***

|                              |                                                                                  |
|------------------------------|----------------------------------------------------------------------------------|
| HRTIM_TIMEVENTLATCH_DISABLED | Event is ignored if it happens during a blank, or passed through during a window |
| HRTIM_TIMEVENTLATCH_ENABLED  | Event is latched and delayed till the end of the blanking or windowing period    |

#### ***HRTIM Timer Fault Enabling***

|                             |                  |
|-----------------------------|------------------|
| HRTIM_TIMFAULTENABLE_NONE   | No fault enabled |
| HRTIM_TIMFAULTENABLE_FAULT1 | Fault 1 enabled  |
| HRTIM_TIMFAULTENABLE_FAULT2 | Fault 2 enabled  |
| HRTIM_TIMFAULTENABLE_FAULT3 | Fault 3 enabled  |
| HRTIM_TIMFAULTENABLE_FAULT4 | Fault 4 enabled  |
| HRTIM_TIMFAULTENABLE_FAULT5 | Fault 5 enabled  |

#### ***HRTIM Timer Fault Lock***

|                              |                                          |
|------------------------------|------------------------------------------|
| HRTIM_TIMFAULTLOCK_READWRITE | Timer fault enabling bits are read/write |
| HRTIM_TIMFAULTLOCK_READONLY  | Timer fault enabling bits are read only  |

#### ***HRTIM Timer identifier***

|                      |                   |
|----------------------|-------------------|
| HRTIM_TIMERID_MASTER | Master identifier |
|----------------------|-------------------|

---

|                       |                    |
|-----------------------|--------------------|
| HRTIM_TIMERID_TIMER_A | Timer A identifier |
| HRTIM_TIMERID_TIMER_B | Timer B identifier |
| HRTIM_TIMERID_TIMER_C | Timer C identifier |
| HRTIM_TIMERID_TIMER_D | Timer D identifier |
| HRTIM_TIMERID_TIMER_E | Timer E identifier |

#### ***HRTIM Timer Index***

|                          |                                             |
|--------------------------|---------------------------------------------|
| HRTIM_TIMERINDEX_TIMER_A | Index used to access timer A registers      |
| HRTIM_TIMERINDEX_TIMER_B | Index used to access timer B registers      |
| HRTIM_TIMERINDEX_TIMER_C | Index used to access timer C registers      |
| HRTIM_TIMERINDEX_TIMER_D | Index used to access timer D registers      |
| HRTIM_TIMERINDEX_TIMER_E | Index used to access timer E registers      |
| HRTIM_TIMERINDEX_MASTER  | Index used to access master registers       |
| HRTIM_TIMERINDEX_COMMON  | Index used to access HRTIM common registers |

#### ***HRTIM Timer Output***

|                  |                               |
|------------------|-------------------------------|
| HRTIM_OUTPUT_TA1 | Timer A - Output 1 identifier |
| HRTIM_OUTPUT_TA2 | Timer A - Output 2 identifier |
| HRTIM_OUTPUT_TB1 | Timer B - Output 1 identifier |
| HRTIM_OUTPUT_TB2 | Timer B - Output 2 identifier |
| HRTIM_OUTPUT_TC1 | Timer C - Output 1 identifier |
| HRTIM_OUTPUT_TC2 | Timer C - Output 2 identifier |
| HRTIM_OUTPUT_TD1 | Timer D - Output 1 identifier |
| HRTIM_OUTPUT_TD2 | Timer D - Output 2 identifier |
| HRTIM_OUTPUT_TE1 | Timer E - Output 1 identifier |
| HRTIM_OUTPUT_TE2 | Timer E - Output 2 identifier |

#### ***HRTIM Timer Push Pull Mode***

|                                |                         |
|--------------------------------|-------------------------|
| HRTIM_TIMPUSHPULLMODE_DISABLED | Push-Pull mode disabled |
| HRTIM_TIMPUSHPULLMODE_ENABLED  | Push-Pull mode enabled  |

#### ***HRTIM Timer Repetition Update***

|                                   |                               |
|-----------------------------------|-------------------------------|
| HRTIM_UPDATEONREPETITION_DISABLED | Update on repetition disabled |
| HRTIM_UPDATEONREPETITION_ENABLED  | Update on repetition enabled  |

#### ***HRTIM Timer Reset Trigger***

|                              |                                                       |
|------------------------------|-------------------------------------------------------|
| HRTIM_TIMRESETTRIGGER_NONE   | No counter reset trigger                              |
| HRTIM_TIMRESETTRIGGER_UPDATE | The timer counter is reset upon update event          |
| HRTIM_TIMRESETTRIGGER_CMP2   | The timer counter is reset upon Timer Compare 2 event |
| HRTIM_TIMRESETTRIGGER_CMP4   | The timer counter is reset upon Timer                 |

|                                   |                                                              |
|-----------------------------------|--------------------------------------------------------------|
|                                   | Compare 4 event                                              |
| HRTIM_TIMRESETTRIGGER_MASTER_PER  | The timer counter is reset upon master timer period event    |
| HRTIM_TIMRESETTRIGGER_MASTER_CMP1 | The timer counter is reset upon master timer Compare 1 event |
| HRTIM_TIMRESETTRIGGER_MASTER_CMP2 | The timer counter is reset upon master timer Compare 2 event |
| HRTIM_TIMRESETTRIGGER_MASTER_CMP3 | The timer counter is reset upon master timer Compare 3 event |
| HRTIM_TIMRESETTRIGGER_MASTER_CMP4 | The timer counter is reset upon master timer Compare 4 event |
| HRTIM_TIMRESETTRIGGER_EEV_1       | The timer counter is reset upon external event 1U            |
| HRTIM_TIMRESETTRIGGER_EEV_2       | The timer counter is reset upon external event 2U            |
| HRTIM_TIMRESETTRIGGER_EEV_3       | The timer counter is reset upon external event 3U            |
| HRTIM_TIMRESETTRIGGER_EEV_4       | The timer counter is reset upon external event 4U            |
| HRTIM_TIMRESETTRIGGER_EEV_5       | The timer counter is reset upon external event 5U            |
| HRTIM_TIMRESETTRIGGER_EEV_6       | The timer counter is reset upon external event 6U            |
| HRTIM_TIMRESETTRIGGER_EEV_7       | The timer counter is reset upon external event 7U            |
| HRTIM_TIMRESETTRIGGER_EEV_8       | The timer counter is reset upon external event 8U            |
| HRTIM_TIMRESETTRIGGER_EEV_9       | The timer counter is reset upon external event 9U            |
| HRTIM_TIMRESETTRIGGER_EEV_10      | The timer counter is reset upon external event 10U           |
| HRTIM_TIMRESETTRIGGER_OTHER1_CMP1 | The timer counter is reset upon other timer Compare 1 event  |
| HRTIM_TIMRESETTRIGGER_OTHER1_CMP2 | The timer counter is reset upon other timer Compare 2 event  |
| HRTIM_TIMRESETTRIGGER_OTHER1_CMP4 | The timer counter is reset upon other timer Compare 4 event  |
| HRTIM_TIMRESETTRIGGER_OTHER2_CMP1 | The timer counter is reset upon other timer Compare 1 event  |
| HRTIM_TIMRESETTRIGGER_OTHER2_CMP2 | The timer counter is reset upon other timer Compare 2 event  |
| HRTIM_TIMRESETTRIGGER_OTHER2_CMP4 | The timer counter is reset upon other timer Compare 4 event  |
| HRTIM_TIMRESETTRIGGER_OTHER3_CMP1 | The timer counter is reset upon other                        |

|                                   |                                                                                      |
|-----------------------------------|--------------------------------------------------------------------------------------|
| HRTIM_TIMRESETTRIGGER_OTHER3_CMP2 | timer Compare 1 event<br>The timer counter is reset upon other timer Compare 2 event |
| HRTIM_TIMRESETTRIGGER_OTHER3_CMP4 | The timer counter is reset upon other timer Compare 4 event                          |
| HRTIM_TIMRESETTRIGGER_OTHER4_CMP1 | The timer counter is reset upon other timer Compare 1 event                          |
| HRTIM_TIMRESETTRIGGER_OTHER4_CMP2 | The timer counter is reset upon other timer Compare 2 event                          |
| HRTIM_TIMRESETTRIGGER_OTHER4_CMP4 | The timer counter is reset upon other timer Compare 4 event                          |

***HRTIM Timer Reset Update***

|                                 |                                              |
|---------------------------------|----------------------------------------------|
| HRTIM_TIMUPDATEONRESET_DISABLED | Update by timer x reset / roll-over disabled |
| HRTIM_TIMUPDATEONRESET_ENABLED  | Update by timer x reset / roll-over enabled  |

***HRTIM Timer Update Trigger***

|                                |                                                         |
|--------------------------------|---------------------------------------------------------|
| HRTIM_TIMUPDATETRIGGER_NONE    | Register update is disabled                             |
| HRTIM_TIMUPDATETRIGGER_MASTER  | Register update is triggered by the master timer update |
| HRTIM_TIMUPDATETRIGGER_TIMER_A | Register update is triggered by the timer A update      |
| HRTIM_TIMUPDATETRIGGER_TIMER_B | Register update is triggered by the timer B update      |
| HRTIM_TIMUPDATETRIGGER_TIMER_C | Register update is triggered by the timer C update      |
| HRTIM_TIMUPDATETRIGGER_TIMER_D | Register update is triggered by the timer D update      |
| HRTIM_TIMUPDATETRIGGER_TIMER_E | Register update is triggered by the timer E update      |

***HRTIM Timing Unit DMA Request Enable***

|                    |                                         |
|--------------------|-----------------------------------------|
| HRTIM_TIM_DMA_NONE | No DMA request enable                   |
| HRTIM_TIM_DMA_CMP1 | Timer compare 1 DMA request enable      |
| HRTIM_TIM_DMA_CMP2 | Timer compare 2 DMA request enable      |
| HRTIM_TIM_DMA_CMP3 | Timer compare 3 DMA request enable      |
| HRTIM_TIM_DMA_CMP4 | Timer compare 4 DMA request enable      |
| HRTIM_TIM_DMA REP  | Timer repetition DMA request enable     |
| HRTIM_TIM_DMA_UPD  | Timer update DMA request enable         |
| HRTIM_TIM_DMA_CPT1 | Timer capture 1 DMA request enable      |
| HRTIM_TIM_DMA_CPT2 | Timer capture 2 DMA request enable      |
| HRTIM_TIM_DMA_SET1 | Timer output 1 set DMA request enable   |
| HRTIM_TIM_DMA_RST1 | Timer output 1 reset DMA request enable |

---

|                                   |                                           |
|-----------------------------------|-------------------------------------------|
| <code>HRTIM_TIM_DMA_SET2</code>   | Timer output 2 set DMA request enable     |
| <code>HRTIM_TIM_DMA_RST2</code>   | Timer output 2 reset DMA request enable   |
| <code>HRTIM_TIM_DMA_RST</code>    | Timer reset DMA request enable            |
| <code>HRTIM_TIM_DMA_DLYPRT</code> | Timer delay protection DMA request enable |

***HRTIM Timing Unit Interrupt Enable***

|                                  |                                         |
|----------------------------------|-----------------------------------------|
| <code>HRTIM_TIM_IT_NONE</code>   | No interrupt enabled                    |
| <code>HRTIM_TIM_IT_CMP1</code>   | Timer compare 1 interrupt enable        |
| <code>HRTIM_TIM_IT_CMP2</code>   | Timer compare 2 interrupt enable        |
| <code>HRTIM_TIM_IT_CMP3</code>   | Timer compare 3 interrupt enable        |
| <code>HRTIM_TIM_IT_CMP4</code>   | Timer compare 4 interrupt enable        |
| <code>HRTIM_TIM_IT REP</code>    | Timer repetition interrupt enable       |
| <code>HRTIM_TIM_IT_UPD</code>    | Timer update interrupt enable           |
| <code>HRTIM_TIM_IT_CPT1</code>   | Timer capture 1 interrupt enable        |
| <code>HRTIM_TIM_IT_CPT2</code>   | Timer capture 2 interrupt enable        |
| <code>HRTIM_TIM_IT_SET1</code>   | Timer output 1 set interrupt enable     |
| <code>HRTIM_TIM_IT_RST1</code>   | Timer output 1 reset interrupt enable   |
| <code>HRTIM_TIM_IT_SET2</code>   | Timer output 2 set interrupt enable     |
| <code>HRTIM_TIM_IT_RST2</code>   | Timer output 2 reset interrupt enable   |
| <code>HRTIM_TIM_IT_RST</code>    | Timer reset interrupt enable            |
| <code>HRTIM_TIM_IT_DLYPRT</code> | Timer delay protection interrupt enable |

***HRTIM Timing Unit Interrupt Flag***

|                                    |                                       |
|------------------------------------|---------------------------------------|
| <code>HRTIM_TIM_FLAG_CMP1</code>   | Timer compare 1 interrupt flag        |
| <code>HRTIM_TIM_FLAG_CMP2</code>   | Timer compare 2 interrupt flag        |
| <code>HRTIM_TIM_FLAG_CMP3</code>   | Timer compare 3 interrupt flag        |
| <code>HRTIM_TIM_FLAG_CMP4</code>   | Timer compare 4 interrupt flag        |
| <code>HRTIM_TIM_FLAG REP</code>    | Timer repetition interrupt flag       |
| <code>HRTIM_TIM_FLAG_UPD</code>    | Timer update interrupt flag           |
| <code>HRTIM_TIM_FLAG_CPT1</code>   | Timer capture 1 interrupt flag        |
| <code>HRTIM_TIM_FLAG_CPT2</code>   | Timer capture 2 interrupt flag        |
| <code>HRTIM_TIM_FLAG_SET1</code>   | Timer output 1 set interrupt flag     |
| <code>HRTIM_TIM_FLAG_RST1</code>   | Timer output 1 reset interrupt flag   |
| <code>HRTIM_TIM_FLAG_SET2</code>   | Timer output 2 set interrupt flag     |
| <code>HRTIM_TIM_FLAG_RST2</code>   | Timer output 2 reset interrupt flag   |
| <code>HRTIM_TIM_FLAG_RST</code>    | Timer reset interrupt flag            |
| <code>HRTIM_TIM_FLAG_DLYPRT</code> | Timer delay protection interrupt flag |

***HRTIM Update Gating***

|                                    |                                                                                                            |
|------------------------------------|------------------------------------------------------------------------------------------------------------|
| HRTIM_UPDATEGATING_INDEPENDENT     | Update done independently from the DMA burst transfer completion                                           |
| HRTIM_UPDATEGATING_DMABURST        | Update done when the DMA burst transfer is completed                                                       |
| HRTIM_UPDATEGATING_DMABURST_UPDATE | Update done on timer roll-over following a DMA burst transfer completion                                   |
| HRTIM_UPDATEGATING_UPDEN1          | Slave timer only - Update done on a rising edge of HRTIM update enable input 1U                            |
| HRTIM_UPDATEGATING_UPDEN2          | Slave timer only - Update done on a rising edge of HRTIM update enable input 2U                            |
| HRTIM_UPDATEGATING_UPDEN3          | Slave timer only - Update done on a rising edge of HRTIM update enable input 3U                            |
| HRTIM_UPDATEGATING_UPDEN1_UPDATE   | Slave timer only - Update done on the update event following a rising edge of HRTIM update enable input 1U |
| HRTIM_UPDATEGATING_UPDEN2_UPDATE   | Slave timer only - Update done on the update event following a rising edge of HRTIM update enable input 2U |
| HRTIM_UPDATEGATING_UPDEN3_UPDATE   | Slave timer only - Update done on the update event following a rising edge of HRTIM update enable input 3U |

## 24 HAL I2C Generic Driver

### 24.1 I2C Firmware driver registers structures

#### 24.1.1 I2C\_InitTypeDef

##### Data Fields

- *uint32\_t Timing*
- *uint32\_t OwnAddress1*
- *uint32\_t AddressingMode*
- *uint32\_t DualAddressMode*
- *uint32\_t OwnAddress2*
- *uint32\_t OwnAddress2Masks*
- *uint32\_t GeneralCallMode*
- *uint32\_t NoStretchMode*

##### Field Documentation

- ***uint32\_t I2C\_InitTypeDef::Timing***  
Specifies the I2C\_TIMINGR\_register value. This parameter calculated by referring to I2C initialization section in Reference manual
- ***uint32\_t I2C\_InitTypeDef::OwnAddress1***  
Specifies the first device own address. This parameter can be a 7-bit or 10-bit address.
- ***uint32\_t I2C\_InitTypeDef::AddressingMode***  
Specifies if 7-bit or 10-bit addressing mode is selected. This parameter can be a value of **I2C\_ADDRESSING\_MODE**
- ***uint32\_t I2C\_InitTypeDef::DualAddressMode***  
Specifies if dual addressing mode is selected. This parameter can be a value of **I2C\_DUAL\_ADDRESSING\_MODE**
- ***uint32\_t I2C\_InitTypeDef::OwnAddress2***  
Specifies the second device own address if dual addressing mode is selected. This parameter can be a 7-bit address.
- ***uint32\_t I2C\_InitTypeDef::OwnAddress2Masks***  
Specifies the acknowledge mask address second device own address if dual addressing mode is selected. This parameter can be a value of **I2C\_OWN\_ADDRESS2\_MASKS**
- ***uint32\_t I2C\_InitTypeDef::GeneralCallMode***  
Specifies if general call mode is selected. This parameter can be a value of **I2C\_GENERAL\_CALL\_ADDRESSING\_MODE**
- ***uint32\_t I2C\_InitTypeDef::NoStretchMode***  
Specifies if nostretch mode is selected. This parameter can be a value of **I2C\_NOSTRETCH\_MODE**

#### 24.1.2 I2C\_HandleTypeDef

##### Data Fields

- *I2C\_TypeDef \* Instance*
- *I2C\_InitTypeDef Init*
- *uint8\_t \* pBuffPtr*
- *uint16\_t XferSize*
- *\_\_IO uint16\_t XferCount*

- `__IO uint32_t XferOptions`
- `__IO uint32_t PreviousState`
- `HAL_StatusTypeDef(* XferISR`
- `DMA_HandleTypeDef * hdmatx`
- `DMA_HandleTypeDef * hdmarx`
- `HAL_LockTypeDef Lock`
- `__IO HAL_I2C_StateTypeDef State`
- `__IO HAL_I2C_ModeTypeDef Mode`
- `__IO uint32_t ErrorCode`
- `__IO uint32_t AddrEventCount`

#### Field Documentation

- `I2C_HandleTypeDef* __I2C_HandleTypeDef::Instance`  
I2C registers base address
- `I2C_InitTypeDef __I2C_HandleTypeDef::Init`  
I2C communication parameters
- `uint8_t* __I2C_HandleTypeDef::pBuffPtr`  
Pointer to I2C transfer buffer
- `uint16_t __I2C_HandleTypeDef::XferSize`  
I2C transfer size
- `__IO uint16_t __I2C_HandleTypeDef::XferCount`  
I2C transfer counter
- `__IO uint32_t __I2C_HandleTypeDef::XferOptions`  
I2C sequential transfer options, this parameter can be a value of `I2C_XFEROPTIONS`
- `__IO uint32_t __I2C_HandleTypeDef::PreviousState`  
I2C communication Previous state
- `HAL_StatusTypeDef(* __I2C_HandleTypeDef::XferISR)(struct __I2C_HandleTypeDef *hi2c, uint32_t ITFlags, uint32_t ITSources)`  
I2C transfer IRQ handler function pointer
- `DMA_HandleTypeDef* __I2C_HandleTypeDef::hdmatx`  
I2C Tx DMA handle parameters
- `DMA_HandleTypeDef* __I2C_HandleTypeDef::hdmarx`  
I2C Rx DMA handle parameters
- `HAL_LockTypeDef __I2C_HandleTypeDef::Lock`  
I2C locking object
- `__IO HAL_I2C_StateTypeDef __I2C_HandleTypeDef::State`  
I2C communication state
- `__IO HAL_I2C_ModeTypeDef __I2C_HandleTypeDef::Mode`  
I2C communication mode
- `__IO uint32_t __I2C_HandleTypeDef::ErrorCode`  
I2C Error code
- `__IO uint32_t __I2C_HandleTypeDef::AddrEventCount`  
I2C Address Event counter

## 24.2 I2C Firmware driver API description

### 24.2.1 How to use this driver

The I2C HAL driver can be used as follows:

1. Declare a `I2C_HandleTypeDef` handle structure, for example: `I2C_HandleTypeDef hi2c;`
2. Initialize the I2C low level resources by implementing the `HAL_I2C_MspInit()` API:
  - a. Enable the I2Cx interface clock

- b. I2C pins configuration
  - Enable the clock for the I2C GPIOs
  - Configure I2C pins as alternate function open-drain
- c. NVIC configuration if you need to use interrupt process
  - Configure the I2Cx interrupt priority
  - Enable the NVIC I2C IRQ Channel
- d. DMA Configuration if you need to use DMA process
  - Declare a DMA\_HandleTypeDef handle structure for the transmit or receive channel
  - Enable the DMAx interface clock using
  - Configure the DMA handle parameters
  - Configure the DMA Tx or Rx channel
  - Associate the initialized DMA handle to the hi2c DMA Tx or Rx handle
  - Configure the priority and enable the NVIC for the transfer complete interrupt on the DMA Tx or Rx channel
3. Configure the Communication Clock Timing, Own Address1, Master Addressing mode, Dual Addressing mode, Own Address2, Own Address2 Mask, General call and Nostretch mode in the hi2c Init structure.
4. Initialize the I2C registers by calling the HAL\_I2C\_Init(), configures also the low level Hardware (GPIO, CLOCK, NVIC...etc) by calling the customized HAL\_I2C\_MspInit(&hi2c) API.
5. To check if target device is ready for communication, use the function HAL\_I2C\_IsDeviceReady()
6. For I2C IO and IO MEM operations, three operation modes are available within this driver :

### **Polling mode IO operation**

- Transmit in master mode an amount of data in blocking mode using HAL\_I2C\_Master\_Transmit()
- Receive in master mode an amount of data in blocking mode using HAL\_I2C\_Master\_Receive()
- Transmit in slave mode an amount of data in blocking mode using HAL\_I2C\_Slave\_Transmit()
- Receive in slave mode an amount of data in blocking mode using HAL\_I2C\_Slave\_Receive()

### **Polling mode IO MEM operation**

- Write an amount of data in blocking mode to a specific memory address using HAL\_I2C\_Mem\_Write()
- Read an amount of data in blocking mode from a specific memory address using HAL\_I2C\_Mem\_Read()

### **Interrupt mode IO operation**

- Transmit in master mode an amount of data in non-blocking mode using HAL\_I2C\_Master\_Transmit\_IT()
- At transmission end of transfer, HAL\_I2C\_MasterTxCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_MasterTxCallback()
- Receive in master mode an amount of data in non-blocking mode using HAL\_I2C\_Master\_Receive\_IT()

- At reception end of transfer, HAL\_I2C\_MasterRxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_MasterRxCpltCallback()
- Transmit in slave mode an amount of data in non-blocking mode using HAL\_I2C\_Slave\_Transmit\_IT()
- At transmission end of transfer, HAL\_I2C\_SlaveTxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_SlaveTxCpltCallback()
- Receive in slave mode an amount of data in non-blocking mode using HAL\_I2C\_Slave\_Receive\_IT()
- At reception end of transfer, HAL\_I2C\_SlaveRxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_SlaveRxCpltCallback()
- In case of transfer Error, HAL\_I2C\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_I2C\_ErrorCallback()
- Abort a master I2C process communication with Interrupt using HAL\_I2C\_Master\_Abort\_IT()
- End of abort process, HAL\_I2C\_AbortCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_AbortCpltCallback()
- Discard a slave I2C process communication using \_\_HAL\_I2C\_GENERATE\_NACK() macro. This action will inform Master to generate a Stop condition to discard the communication.

### Interrupt mode IO sequential operation



These interfaces allow to manage a sequential transfer with a repeated start condition when a direction change during transfer

- A specific option field manage the different steps of a sequential transfer
- Option field values are defined through @ref I2C\_XFEROPTIONS and are listed below:
  - I2C\_FIRST\_AND\_LAST\_FRAME: No sequential usage, functionnal is same as associated interfaces in no sequential mode
  - I2C\_FIRST\_FRAME: Sequential usage, this option allow to manage a sequence with start condition, address and data to transfer without a final stop condition
  - I2C\_FIRST\_AND\_NEXT\_FRAME: Sequential usage (Master only), this option allow to manage a sequence with start condition, address and data to transfer without a final stop condition, an then permit a call the same master sequential interface several times (like HAL\_I2C\_Master\_Sequential\_Transmit\_IT() then HAL\_I2C\_Master\_Sequential\_Transmit\_IT())
  - I2C\_NEXT\_FRAME: Sequential usage, this option allow to manage a sequence with a restart condition, address and with new data to transfer if the direction change or manage only the new data to transfer if no direction change and without a final stop condition in both cases
  - I2C\_LAST\_FRAME: Sequential usage, this option allow to manage a sequence with a restart condition, address and with new data to transfer if the direction change or manage only the new data to transfer if no direction change and with a final stop condition in both cases
- Differents sequential I2C interfaces are listed below:
  - Sequential transmit in master I2C mode an amount of data in non-blocking mode using HAL\_I2C\_Master\_Sequential\_Transmit\_IT()

- At transmission end of current frame transfer,  
HAL\_I2C\_MasterTxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_MasterTxCpltCallback()
- Sequential receive in master I2C mode an amount of data in non-blocking mode using HAL\_I2C\_Master\_Sequential\_Receive\_IT()
- At reception end of current frame transfer,  
HAL\_I2C\_MasterRxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_MasterRxCpltCallback()
- Abort a master I2C process communication with Interrupt using  
HAL\_I2C\_Master\_Abort\_IT()
  - End of abort process, HAL\_I2C\_AbortCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_AbortCpltCallback()
- Enable/disable the Address listen mode in slave I2C mode using  
HAL\_I2C\_EnableListen\_IT() HAL\_I2C\_DisableListen\_IT()
  - When address slave I2C match, HAL\_I2C\_AddrCallback() is executed and user can add his own code to check the Address Match Code and the transmission direction request by master (Write/Read).
  - At Listen mode end HAL\_I2C\_ListenCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_ListenCpltCallback()
- Sequential transmit in slave I2C mode an amount of data in non-blocking mode using HAL\_I2C\_Slave\_Sequential\_Transmit\_IT()
  - At transmission end of current frame transfer,  
HAL\_I2C\_SlaveTxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_SlaveTxCpltCallback()
- Sequential receive in slave I2C mode an amount of data in non-blocking mode using HAL\_I2C\_Slave\_Sequential\_Receive\_IT()
  - At reception end of current frame transfer, HAL\_I2C\_SlaveRxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_SlaveRxCpltCallback()
- In case of transfer Error, HAL\_I2C\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_I2C\_ErrorCallback()
- Abort a master I2C process communication with Interrupt using  
HAL\_I2C\_Master\_Abort\_IT()
  - End of abort process, HAL\_I2C\_AbortCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_AbortCpltCallback()
- Discard a slave I2C process communication using  
\_\_HAL\_I2C\_GENERATE\_NACK() macro. This action will inform Master to generate a Stop condition to discard the communication.

### Interrupt mode IO MEM operation

- Write an amount of data in non-blocking mode with Interrupt to a specific memory address using HAL\_I2C\_Mem\_Write\_IT()
- At Memory end of write transfer, HAL\_I2C\_MemTxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_MemTxCpltCallback()
- Read an amount of data in non-blocking mode with Interrupt from a specific memory address using HAL\_I2C\_Mem\_Read\_IT()
- At Memory end of read transfer, HAL\_I2C\_MemRxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_MemRxCpltCallback()

- In case of transfer Error, HAL\_I2C\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_I2C\_ErrorCallback()

### DMA mode IO operation

- Transmit in master mode an amount of data in non-blocking mode (DMA) using HAL\_I2C\_Master\_Transmit\_DMA()
- At transmission end of transfer, HAL\_I2C\_MasterTxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_MasterTxCpltCallback()
- Receive in master mode an amount of data in non-blocking mode (DMA) using HAL\_I2C\_Master\_Receive\_DMA()
- At reception end of transfer, HAL\_I2C\_MasterRxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_MasterRxCpltCallback()
- Transmit in slave mode an amount of data in non-blocking mode (DMA) using HAL\_I2C\_Slave\_Transmit\_DMA()
- At transmission end of transfer, HAL\_I2C\_SlaveTxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_SlaveTxCpltCallback()
- Receive in slave mode an amount of data in non-blocking mode (DMA) using HAL\_I2C\_Slave\_Receive\_DMA()
- At reception end of transfer, HAL\_I2C\_SlaveRxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_SlaveRxCpltCallback()
- In case of transfer Error, HAL\_I2C\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_I2C\_ErrorCallback()
- Abort a master I2C process communication with Interrupt using HAL\_I2C\_Master\_Abort\_IT()
- End of abort process, HAL\_I2C\_AbortCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_AbortCpltCallback()
- Discard a slave I2C process communication using \_\_HAL\_I2C\_GENERATE\_NACK() macro. This action will inform Master to generate a Stop condition to discard the communication.

### DMA mode IO MEM operation

- Write an amount of data in non-blocking mode with DMA to a specific memory address using HAL\_I2C\_Mem\_Write\_DMA()
- At Memory end of write transfer, HAL\_I2C\_MemTxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_MemTxCpltCallback()
- Read an amount of data in non-blocking mode with DMA from a specific memory address using HAL\_I2C\_Mem\_Read\_DMA()
- At Memory end of read transfer, HAL\_I2C\_MemRxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_I2C\_MemRxCpltCallback()
- In case of transfer Error, HAL\_I2C\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_I2C\_ErrorCallback()

### I2C HAL driver macros list

Below the list of most used macros in I2C HAL driver.

- \_\_HAL\_I2C\_ENABLE: Enable the I2C peripheral
- \_\_HAL\_I2C\_DISABLE: Disable the I2C peripheral

- `__HAL_I2C_GENERATE_NACK`: Generate a Non-Acknowledge I2C peripheral in Slave mode
- `__HAL_I2C_GET_FLAG`: Check whether the specified I2C flag is set or not
- `__HAL_I2C_CLEAR_FLAG`: Clear the specified I2C pending flag
- `__HAL_I2C_ENABLE_IT`: Enable the specified I2C interrupt
- `__HAL_I2C_DISABLE_IT`: Disable the specified I2C interrupt



You can refer to the I2C HAL driver header file for more useful macros

## 24.2.2 Initialization and de-initialization functions

This subsection provides a set of functions allowing to initialize and deinitialize the I2Cx peripheral:

- User must Implement `HAL_I2C_MspInit()` function in which he configures all related peripherals resources (CLOCK, GPIO, DMA, IT and NVIC ).
- Call the function `HAL_I2C_Init()` to configure the selected device with the selected configuration:
  - Clock Timing
  - Own Address 1
  - Addressing mode (Master, Slave)
  - Dual Addressing mode
  - Own Address 2
  - Own Address 2 Mask
  - General call mode
  - Nostretch mode
- Call the function `HAL_I2C_DelInit()` to restore the default configuration of the selected I2Cx peripheral.

This section contains the following APIs:

- [`HAL\_I2C\_Init\(\)`](#)
- [`HAL\_I2C\_DelInit\(\)`](#)
- [`HAL\_I2C\_MspInit\(\)`](#)
- [`HAL\_I2C\_MspDelInit\(\)`](#)

## 24.2.3 IO operation functions

This subsection provides a set of functions allowing to manage the I2C data transfers.

1. There are two modes of transfer:
  - Blocking mode : The communication is performed in the polling mode. The status of all data processing is returned by the same function after finishing transfer.
  - No-Blocking mode : The communication is performed using Interrupts or DMA. These functions return the status of the transfer startup. The end of the data processing will be indicated through the dedicated I2C IRQ when using Interrupt mode or the DMA IRQ when using DMA mode.
2. Blocking mode functions are :
  - `HAL_I2C_Master_Transmit()`
  - `HAL_I2C_Master_Receive()`
  - `HAL_I2C_Slave_Transmit()`
  - `HAL_I2C_Slave_Receive()`
  - `HAL_I2C_Mem_Write()`

- HAL\_I2C\_Mem\_Read()
  - HAL\_I2C\_IsDeviceReady()
3. No-Blocking mode functions with Interrupt are :
- HAL\_I2C\_Master\_Transmit\_IT()
  - HAL\_I2C\_Master\_Receive\_IT()
  - HAL\_I2C\_Slave\_Transmit\_IT()
  - HAL\_I2C\_Slave\_Receive\_IT()
  - HAL\_I2C\_Mem\_Write\_IT()
  - HAL\_I2C\_Mem\_Read\_IT()
4. No-Blocking mode functions with DMA are :
- HAL\_I2C\_Master\_Transmit\_DMA()
  - HAL\_I2C\_Master\_Receive\_DMA()
  - HAL\_I2C\_Slave\_Transmit\_DMA()
  - HAL\_I2C\_Slave\_Receive\_DMA()
  - HAL\_I2C\_Mem\_Write\_DMA()
  - HAL\_I2C\_Mem\_Read\_DMA()
5. A set of Transfer Complete Callbacks are provided in non Blocking mode:
- HAL\_I2C\_MemTxCpltCallback()
  - HAL\_I2C\_MemRxCpltCallback()
  - HAL\_I2C\_MasterTxCpltCallback()
  - HAL\_I2C\_MasterRxCpltCallback()
  - HAL\_I2C\_SlaveTxCpltCallback()
  - HAL\_I2C\_SlaveRxCpltCallback()
  - HAL\_I2C\_ErrorCallback()

This section contains the following APIs:

- [\*\*HAL\\_I2C\\_Master\\_Transmit\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Master\\_Receive\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Slave\\_Transmit\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Slave\\_Receive\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Master\\_Transmit\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Master\\_Receive\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Slave\\_Transmit\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Slave\\_Receive\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Master\\_Transmit\\_DMA\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Master\\_Receive\\_DMA\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Slave\\_Transmit\\_DMA\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Slave\\_Receive\\_DMA\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Mem\\_Write\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Mem\\_Read\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Mem\\_Write\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Mem\\_Read\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Mem\\_Write\\_DMA\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Mem\\_Read\\_DMA\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_IsDeviceReady\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Master\\_Sequential\\_Transmit\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Master\\_Sequential\\_Receive\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Slave\\_Sequential\\_Transmit\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Slave\\_Sequential\\_Receive\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_EnableListen\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_DisableListen\\_IT\(\)\*\*](#)
- [\*\*HAL\\_I2C\\_Master\\_Abort\\_IT\(\)\*\*](#)

## 24.2.4 Peripheral State, Mode and Error functions

This subsection permit to get in run-time the status of the peripheral and the data flow.

This section contains the following APIs:

- [\*\*\*HAL\\_I2C\\_GetState\(\)\*\*\*](#)
- [\*\*\*HAL\\_I2C\\_GetMode\(\)\*\*\*](#)
- [\*\*\*HAL\\_I2C\\_GetError\(\)\*\*\*](#)

## 24.2.5 Detailed description of functions

### **HAL\_I2C\_Init**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Init (I2C_HandleTypeDef * hi2c)</b>                                                                                                             |
| Function description | Initializes the I2C according to the specified parameters in the I2C_InitTypeDef and initialize the associated handle.                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                       |

### **HAL\_I2C\_DelInit**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_DelInit (I2C_HandleTypeDef * hi2c)</b>                                                                                                          |
| Function description | DeInitialize the I2C peripheral.                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                       |

### **HAL\_I2C\_MspInit**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2C_MspInit (I2C_HandleTypeDef * hi2c)</b>                                                                                                                       |
| Function description | Initialize the I2C MSP.                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

### **HAL\_I2C\_MspDelInit**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2C_MspDelInit (I2C_HandleTypeDef * hi2c)</b>                                                                                                                    |
| Function description | DeInitialize the I2C MSP.                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_I2C\_Master\_Transmit**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Master_Transmit<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t *<br/>pData, uint16_t Size, uint32_t Timeout)</b>                                                                                                                                                                                                                                                                                                                |
| Function description | Transmits in master mode an amount of data in blocking mode.                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                    |

**HAL\_I2C\_Master\_Receive**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Master_Receive<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t *<br/>pData, uint16_t Size, uint32_t Timeout)</b>                                                                                                                                                                                                                                                                                                                 |
| Function description | Receives in master mode an amount of data in blocking mode.                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                    |

**HAL\_I2C\_Slave\_Transmit**

|                      |                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Slave_Transmit<br/>(I2C_HandleTypeDef * hi2c, uint8_t * pData, uint16_t Size,<br/>uint32_t Timeout)</b>                                                                                                                                                                                       |
| Function description | Transmits in slave mode an amount of data in blocking mode.                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                     |

**HAL\_I2C\_Slave\_Receive**

|                      |                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Slave_Receive<br/>(I2C_HandleTypeDef * hi2c, uint8_t * pData, uint16_t Size,<br/>uint32_t Timeout)</b>                                                                                                                                                                                        |
| Function description | Receive in slave mode an amount of data in blocking mode.                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                     |

**HAL\_I2C\_Mem\_Write**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Mem_Write<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint16_t<br/>MemAddress, uint16_t MemAddSize, uint8_t * pData, uint16_t<br/>Size, uint32_t Timeout)</b>                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Write an amount of data in blocking mode to a specific memory address.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>MemAddress:</b> Internal memory address</li> <li>• <b>MemAddSize:</b> Size of internal memory address</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

**HAL\_I2C\_Mem\_Read**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Mem_Read<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint16_t<br/>MemAddress, uint16_t MemAddSize, uint8_t * pData, uint16_t<br/>Size, uint32_t Timeout)</b>                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Read an amount of data in blocking mode from a specific memory address.                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>MemAddress:</b> Internal memory address</li> <li>• <b>MemAddSize:</b> Size of internal memory address</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> </ul> |

- **Timeout:** Timeout duration
- Return values
- **HAL:** status

### **HAL\_I2C\_IsDeviceReady**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_IsDeviceReady<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint32_t Trials, uint32_t Timeout)</b>                                                                                                                                                                                                                                                                              |
| Function description | Checks if target device is ready for communication.                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>Trials:</b> Number of trials</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                               |
| Notes                | • This function is used with Memory devices                                                                                                                                                                                                                                                                                                                                                                        |

### **HAL\_I2C\_Master\_Transmit\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Master_Transmit_IT<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                                                                                                                                                       |
| Function description | Transmit in master mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                          |

### **HAL\_I2C\_Master\_Receive\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Master_Receive_IT<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                                                                                                      |
| Function description | Receive in master mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>pData:</b> Pointer to data buffer</li> </ul> |

- **Size:** Amount of data to be sent
- Return values      • **HAL:** status

### **HAL\_I2C\_Slave\_Transmit\_IT**

|                      |                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Slave_Transmit_IT<br/>(I2C_HandleTypeDef * hi2c, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                              |
| Function description | Transmit in slave mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                           |

### **HAL\_I2C\_Slave\_Receive\_IT**

|                      |                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Slave_Receive_IT<br/>(I2C_HandleTypeDef * hi2c, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                               |
| Function description | Receive in slave mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                           |

### **HAL\_I2C\_Mem\_Write\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Mem_Write_IT<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint16_t MemAddress, uint16_t MemAddSize, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Write an amount of data in non-blocking mode with Interrupt to a specific memory address.                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>MemAddress:</b> Internal memory address</li> <li>• <b>MemAddSize:</b> Size of internal memory address</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

**HAL\_I2C\_Mem\_Read\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_I2C_Mem_Read_IT<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint16_t<br/>MemAddress, uint16_t MemAddSize, uint8_t * pData, uint16_t<br/>Size)</code>                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Read an amount of data in non-blocking mode with Interrupt from a specific memory address.                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>MemAddress:</b> Internal memory address</li> <li>• <b>MemAddSize:</b> Size of internal memory address</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**HAL\_I2C\_Master\_Sequential\_Transmit\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef<br/>HAL_I2C_Master_Sequential_Transmit_IT<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t *<br/>pData, uint16_t Size, uint32_t XferOptions)</code>                                                                                                                                                                                                                                                                                                                                     |
| Function description | Sequential transmit in master I2C mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>XferOptions:</b> Options of Transfer, value of I2C Sequential Transfer Options</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This interface allow to manage repeated start condition when a direction change during transfer</li> </ul>                                                                                                                                                                                                                                                                                                                                                                        |

**HAL\_I2C\_Master\_Sequential\_Receive\_IT**

|                      |                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_I2C_Master_Sequential_Receive_IT<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t *<br/>pData, uint16_t Size, uint32_t XferOptions)</code>                                                                   |
| Function description | Sequential receive in master I2C mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits</li> </ul> |

---

|                                                                                     |                                                                                                   |
|-------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------|
|                                                                                     | address value in datasheet must be shift at right before call interface                           |
| • <b>pData:</b> Pointer to data buffer                                              |                                                                                                   |
| • <b>Size:</b> Amount of data to be sent                                            |                                                                                                   |
| • <b>XferOptions:</b> Options of Transfer, value of I2C Sequential Transfer Options |                                                                                                   |
| Return values                                                                       | • <b>HAL:</b> status                                                                              |
| Notes                                                                               | • This interface allow to manage repeated start condition when a direction change during transfer |

### HAL\_I2C\_Slave\_Sequential\_Transmit\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Slave_Sequential_Transmit_IT(I2C_HandleTypeDef * hi2c, uint8_t * pData, uint16_t Size, uint32_t XferOptions)</b>                                                                                                                                                                                                                               |
| Function description | Sequential transmit in slave/device I2C mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>XferOptions:</b> Options of Transfer, value of I2C Sequential Transfer Options</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                        |
| Notes                | • This interface allow to manage repeated start condition when a direction change during transfer                                                                                                                                                                                                                                                                           |

### HAL\_I2C\_Slave\_Sequential\_Receive\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Slave_Sequential_Receive_IT(I2C_HandleTypeDef * hi2c, uint8_t * pData, uint16_t Size, uint32_t XferOptions)</b>                                                                                                                                                                                                                                |
| Function description | Sequential receive in slave/device I2C mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>XferOptions:</b> Options of Transfer, value of I2C Sequential Transfer Options</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                        |
| Notes                | • This interface allow to manage repeated start condition when a direction change during transfer                                                                                                                                                                                                                                                                           |

**HAL\_I2C\_EnableListen\_IT**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_EnableListen_IT<br/>(I2C_HandleTypeDef * hi2c)</b>                                                                                              |
| Function description | Enable the Address listen mode with Interrupt.                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                       |

**HAL\_I2C\_DisableListen\_IT**

|                      |                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_DisableListen_IT<br/>(I2C_HandleTypeDef * hi2c)</b>                                                                                            |
| Function description | Disable the Address listen mode with Interrupt.                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                      |

**HAL\_I2C\_Master\_Abort\_IT**

|                      |                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Master_Abort_IT<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress)</b>                                                                                                                                                                                                                        |
| Function description | Abort a master I2C IT or DMA process communication with Interrupt.                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                      |

**HAL\_I2C\_Master\_Transmit\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Master_Transmit_DMA<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                                                                                                                                                      |
| Function description | Transmit in master mode an amount of data in non-blocking mode with DMA.                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                        |

**HAL\_I2C\_Master\_Receive\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Master_Receive_DMA<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t *<br/>pData, uint16_t Size)</b>                                                                                                                                                                                                                                                                                   |
| Function description | Receive in master mode an amount of data in non-blocking mode with DMA.                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                        |

**HAL\_I2C\_Slave\_Transmit\_DMA**

|                      |                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Slave_Transmit_DMA<br/>(I2C_HandleTypeDef * hi2c, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                             |
| Function description | Transmit in slave mode an amount of data in non-blocking mode with DMA.                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                         |

**HAL\_I2C\_Slave\_Receive\_DMA**

|                      |                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Slave_Receive_DMA<br/>(I2C_HandleTypeDef * hi2c, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                              |
| Function description | Receive in slave mode an amount of data in non-blocking mode with DMA.                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                         |

**HAL\_I2C\_Mem\_Write\_DMA**

|                      |                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Mem_Write_DMA<br/>(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint16_t<br/>MemAddress, uint16_t MemAddSize, uint8_t * pData, uint16_t<br/>Size)</b> |
| Function description | Write an amount of data in non-blocking mode with DMA to a specific memory address.                                                                                                  |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li><b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li><b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li><b>MemAddress:</b> Internal memory address</li> <li><b>MemAddSize:</b> Size of internal memory address</li> <li><b>pData:</b> Pointer to data buffer</li> <li><b>Size:</b> Amount of data to be sent</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

### HAL\_I2C\_Mem\_Read\_DMA

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2C_Mem_Read_DMA(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint16_t MemAddress, uint16_t MemAddSize, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Reads an amount of data in non-blocking mode with DMA from a specific memory address.                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li><b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li><b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li><b>MemAddress:</b> Internal memory address</li> <li><b>MemAddSize:</b> Size of internal memory address</li> <li><b>pData:</b> Pointer to data buffer</li> <li><b>Size:</b> Amount of data to be read</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

### HAL\_I2C\_EV\_IRQHandler

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2C_EV_IRQHandler(I2C_HandleTypeDef * hi2c)</b>                                                                                                                |
| Function description | This function handles I2C event interrupt request.                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li><b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                              |

### HAL\_I2C\_ER\_IRQHandler

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2C_ER_IRQHandler(I2C_HandleTypeDef * hi2c)</b>                                                                                                                |
| Function description | This function handles I2C error interrupt request.                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li><b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                              |

**HAL\_I2C\_MasterTxCpltCallback**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2C_MasterTxCpltCallback (I2C_HandleTypeDef * hi2c)</b>                                                                                                          |
| Function description | Master Tx Transfer completed callback.                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_I2C\_MasterRxCpltCallback**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2C_MasterRxCpltCallback (I2C_HandleTypeDef * hi2c)</b>                                                                                                          |
| Function description | Master Rx Transfer completed callback.                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_I2C\_SlaveTxCpltCallback**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2C_SlaveTxCpltCallback (I2C_HandleTypeDef * hi2c)</b>                                                                                                           |
| Function description | Slave Tx Transfer completed callback.                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_I2C\_SlaveRxCpltCallback**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2C_SlaveRxCpltCallback (I2C_HandleTypeDef * hi2c)</b>                                                                                                           |
| Function description | Slave Rx Transfer completed callback.                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_I2C\_AddrCallback**

|                      |                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2C_AddrCallback (I2C_HandleTypeDef * hi2c, uint8_t TransferDirection, uint16_t AddrMatchCode)</b>                                                                                                                                                                                                                                                             |
| Function description | Slave Address Match callback.                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> <li>• <b>TransferDirection:</b> Master request Transfer Direction (Write/Read), value of I2C Transfer Direction Master Point of View</li> <li>• <b>AddrMatchCode:</b> Address Match Code</li> </ul> |

---

|               |                                                               |
|---------------|---------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"><li>• <b>None</b></li></ul> |
|---------------|---------------------------------------------------------------|

**HAL\_I2C\_ListenCpltCallback**

Function name      **void HAL\_I2C\_ListenCpltCallback (I2C\_HandleTypeDef \* hi2c)**

Function description      Listen Complete callback.

Parameters      

- **hi2c:** Pointer to a I2C\_HandleTypeDef structure that contains the configuration information for the specified I2C.

|               |                                                               |
|---------------|---------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"><li>• <b>None</b></li></ul> |
|---------------|---------------------------------------------------------------|

**HAL\_I2C\_MemTxCpltCallback**

Function name      **void HAL\_I2C\_MemTxCpltCallback (I2C\_HandleTypeDef \* hi2c)**

Function description      Memory Tx Transfer completed callback.

Parameters      

- **hi2c:** Pointer to a I2C\_HandleTypeDef structure that contains the configuration information for the specified I2C.

|               |                                                               |
|---------------|---------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"><li>• <b>None</b></li></ul> |
|---------------|---------------------------------------------------------------|

**HAL\_I2C\_MemRxCpltCallback**

Function name      **void HAL\_I2C\_MemRxCpltCallback (I2C\_HandleTypeDef \* hi2c)**

Function description      Memory Rx Transfer completed callback.

Parameters      

- **hi2c:** Pointer to a I2C\_HandleTypeDef structure that contains the configuration information for the specified I2C.

|               |                                                               |
|---------------|---------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"><li>• <b>None</b></li></ul> |
|---------------|---------------------------------------------------------------|

**HAL\_I2C\_ErrorCallback**

Function name      **void HAL\_I2C\_ErrorCallback (I2C\_HandleTypeDef \* hi2c)**

Function description      I2C error callback.

Parameters      

- **hi2c:** Pointer to a I2C\_HandleTypeDef structure that contains the configuration information for the specified I2C.

|               |                                                               |
|---------------|---------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"><li>• <b>None</b></li></ul> |
|---------------|---------------------------------------------------------------|

**HAL\_I2C\_AbortCpltCallback**

Function name      **void HAL\_I2C\_AbortCpltCallback (I2C\_HandleTypeDef \* hi2c)**

Function description      I2C abort callback.

Parameters      

- **hi2c:** Pointer to a I2C\_HandleTypeDef structure that contains the configuration information for the specified I2C.

|               |                                                               |
|---------------|---------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"><li>• <b>None</b></li></ul> |
|---------------|---------------------------------------------------------------|

**HAL\_I2C\_GetState**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_I2C_StateTypeDef HAL_I2C_GetState<br/>(I2C_HandleTypeDef * hi2c)</b>                                                                                                  |
| Function description | Return the I2C handle state.                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                        |

**HAL\_I2C\_GetMode**

|                      |                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_I2C_ModeTypeDef HAL_I2C_GetMode<br/>(I2C_HandleTypeDef * hi2c)</b>                                                                                            |
| Function description | Returns the I2C Master, Slave, Memory or no mode.                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for I2C module</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> mode</li> </ul>                                                                                                 |

**HAL\_I2C\_GetError**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_I2C_GetError (I2C_HandleTypeDef * hi2c)</b>                                                                                                                  |
| Function description | Return the I2C error code.                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2c:</b> Pointer to a I2C_HandleTypeDef structure that contains the configuration information for the specified I2C.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>I2C:</b> Error Code</li> </ul>                                                                                                   |

## 24.3 I2C Firmware driver defines

### 24.3.1 I2C

*I2C Addressing Mode*

I2C\_ADDRESSINGMODE\_7BIT  
I2C\_ADDRESSINGMODE\_10BIT

*I2C Dual Addressing Mode*

I2C\_DUALADDRESS\_DISABLE  
I2C\_DUALADDRESS\_ENABLE

*I2C Error Code definition*

|                           |                    |
|---------------------------|--------------------|
| <b>HAL_I2C_ERROR_NONE</b> | No error           |
| <b>HAL_I2C_ERROR_BERR</b> | BERR error         |
| <b>HAL_I2C_ERROR_ARLO</b> | ARLO error         |
| <b>HAL_I2C_ERROR_AF</b>   | ACKF error         |
| <b>HAL_I2C_ERROR_OVR</b>  | OVR error          |
| <b>HAL_I2C_ERROR_DMA</b>  | DMA transfer error |

|                       |                       |
|-----------------------|-----------------------|
| HAL_I2C_ERROR_TIMEOUT | Timeout error         |
| HAL_I2C_ERROR_SIZE    | Size Management error |

**I2C Exported Macros**

|                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>__HAL_I2C_RESET_HANDLE_STATE</u> | <b>Description:</b> <ul style="list-style-type: none"><li>Reset I2C handle state.</li></ul> <b>Parameters:</b> <ul style="list-style-type: none"><li><u>__HANDLE__</u>: specifies the I2C Handle.</li></ul> <b>Return value:</b> <ul style="list-style-type: none"><li>None</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <u>__HAL_I2C_ENABLE_IT</u>          | <b>Description:</b> <ul style="list-style-type: none"><li>Enable the specified I2C interrupt.</li></ul> <b>Parameters:</b> <ul style="list-style-type: none"><li><u>__HANDLE__</u>: specifies the I2C Handle.</li><li><u>__INTERRUPT__</u>: specifies the interrupt source to enable. This parameter can be one of the following values:<ul style="list-style-type: none"><li>I2C_IT_ERRI Errors interrupt enable</li><li>I2C_IT_TCI Transfer complete interrupt enable</li><li>I2C_IT_STOPI STOP detection interrupt enable</li><li>I2C_IT_NACKI NACK received interrupt enable</li><li>I2C_IT_ADDRI Address match interrupt enable</li><li>I2C_IT_RXI RX interrupt enable</li><li>I2C_IT_TXI TX interrupt enable</li></ul></li></ul> <b>Return value:</b> <ul style="list-style-type: none"><li>None</li></ul> |
| <u>__HAL_I2C_DISABLE_IT</u>         | <b>Description:</b> <ul style="list-style-type: none"><li>Disable the specified I2C interrupt.</li></ul> <b>Parameters:</b> <ul style="list-style-type: none"><li><u>__HANDLE__</u>: specifies the I2C Handle.</li><li><u>__INTERRUPT__</u>: specifies the interrupt source to disable. This parameter can be one of the following values:<ul style="list-style-type: none"><li>I2C_IT_ERRI Errors interrupt enable</li><li>I2C_IT_TCI Transfer complete interrupt enable</li><li>I2C_IT_STOPI STOP detection interrupt enable</li><li>I2C_IT_NACKI NACK received interrupt enable</li><li>I2C_IT_ADDRI Address match interrupt enable</li></ul></li></ul>                                                                                                                                                       |

- I2C\_IT\_RXI RX interrupt enable
- I2C\_IT\_TXI TX interrupt enable

**Return value:**

- None

**\_HAL\_I2C\_GET\_IT\_SOURCE**

- Check whether the specified I2C interrupt source is enabled or not.

**Parameters:**

- \_HANDLE\_: specifies the I2C Handle.
- \_INTERRUPT\_: specifies the I2C interrupt source to check. This parameter can be one of the following values:
  - I2C\_IT\_ERRI Errors interrupt enable
  - I2C\_IT\_TCI Transfer complete interrupt enable
  - I2C\_IT\_STOPI STOP detection interrupt enable
  - I2C\_IT\_NACKI NACK received interrupt enable
  - I2C\_IT\_ADDRI Address match interrupt enable
  - I2C\_IT\_RXI RX interrupt enable
  - I2C\_IT\_TXI TX interrupt enable

**Return value:**

- The new state of \_INTERRUPT\_ (SET or RESET).

**\_HAL\_I2C\_GET\_FLAG**

- Check whether the specified I2C flag is set or not.

**Parameters:**

- \_HANDLE\_: specifies the I2C Handle.
- \_FLAG\_: specifies the flag to check. This parameter can be one of the following values:
  - I2C\_FLAG\_TXE Transmit data register empty
  - I2C\_FLAG\_TXIS Transmit interrupt status
  - I2C\_FLAG\_RXNE Receive data register not empty
  - I2C\_FLAG\_ADDR Address matched (slave mode)
  - I2C\_FLAG\_AF Acknowledge failure received flag
  - I2C\_FLAG\_STOPF STOP detection flag
  - I2C\_FLAG\_TC Transfer complete (master mode)

- I2C\_FLAG\_TCR Transfer complete reload
- I2C\_FLAG\_BERR Bus error
- I2C\_FLAG\_ARLO Arbitration lost
- I2C\_FLAG\_OVR Overrun/Underrun
- I2C\_FLAG\_PECERR PEC error in reception
- I2C\_FLAG\_TIMEOUT Timeout or Tlow detection flag
- I2C\_FLAG\_ALERT SMBus alert
- I2C\_FLAG\_BUSY Bus busy
- I2C\_FLAG\_DIR Transfer direction (slave mode)

**Return value:**

- The new state of \_\_FLAG\_\_ (SET or RESET).

[\\_\\_HAL\\_I2C\\_CLEAR\\_FLAG](#)**Description:**

- Clear the I2C pending flags which are cleared by writing 1 in a specific bit.

**Parameters:**

- \_\_HANDLE\_\_: specifies the I2C Handle.
- \_\_FLAG\_\_: specifies the flag to clear. This parameter can be any combination of the following values:
  - I2C\_FLAG\_TXE Transmit data register empty
  - I2C\_FLAG\_ADDR Address matched (slave mode)
  - I2C\_FLAG\_AF Acknowledge failure received flag
  - I2C\_FLAG\_STOPF STOP detection flag
  - I2C\_FLAG\_BERR Bus error
  - I2C\_FLAG\_ARLO Arbitration lost
  - I2C\_FLAG\_OVR Overrun/Underrun
  - I2C\_FLAG\_PECERR PEC error in reception
  - I2C\_FLAG\_TIMEOUT Timeout or Tlow detection flag
  - I2C\_FLAG\_ALERT SMBus alert

**Return value:**

- None

[\\_\\_HAL\\_I2C\\_ENABLE](#)**Description:**

- Enable the specified I2C peripheral.

**Parameters:**

- \_\_HANDLE\_\_: specifies the I2C Handle.

**Return value:**

- None

`__HAL_I2C_DISABLE`

**Description:**

- Disable the specified I2C peripheral.

**Parameters:**

- `__HANDLE__`: specifies the I2C Handle.

**Return value:**

- None

`__HAL_I2C_GENERATE_NACK`

**Description:**

- Generate a Non-Acknowledge I2C peripheral in Slave mode.

**Parameters:**

- `__HANDLE__`: specifies the I2C Handle.

**Return value:**

- None

***I2C Flag definition***

`I2C_FLAG_TXE`

`I2C_FLAG_TXIS`

`I2C_FLAG_RXNE`

`I2C_FLAG_ADDR`

`I2C_FLAG_AF`

`I2C_FLAG_STOPF`

`I2C_FLAG_TC`

`I2C_FLAG_TCR`

`I2C_FLAG_BERR`

`I2C_FLAG_ARLO`

`I2C_FLAG_OVR`

`I2C_FLAG_PECERR`

`I2C_FLAG_TIMEOUT`

`I2C_FLAG_ALERT`

`I2C_FLAG_BUSY`

`I2C_FLAG_DIR`

***I2C General Call Addressing Mode***

`I2C_GENERALCALL_DISABLE`

`I2C_GENERALCALL_ENABLE`

***I2C Interrupt configuration definition***

`I2C_IT_ERRI`

I2C\_IT\_TCI  
I2C\_IT\_STOPI  
I2C\_IT\_NACKI  
I2C\_IT\_ADDRI  
I2C\_IT\_RXI  
I2C\_IT\_TXI

***I2C Memory Address Size***

I2C\_MEMADD\_SIZE\_8BIT  
I2C\_MEMADD\_SIZE\_16BIT

***I2C No-Stretch Mode***

I2C\_NOSTRETCH\_DISABLE  
I2C\_NOSTRETCH\_ENABLE

***I2C Own Address2 Masks***

I2C\_OA2\_NOMASK  
I2C\_OA2\_MASK01  
I2C\_OA2\_MASK02  
I2C\_OA2\_MASK03  
I2C\_OA2\_MASK04  
I2C\_OA2\_MASK05  
I2C\_OA2\_MASK06  
I2C\_OA2\_MASK07

***I2C Reload End Mode***

I2C\_RELOAD\_MODE  
I2C\_AUTOEND\_MODE  
I2C\_SOFTEND\_MODE

***I2C Start or Stop Mode***

I2C\_NO\_STARTSTOP  
I2C\_GENERATE\_STOP  
I2C\_GENERATE\_START\_READ  
I2C\_GENERATE\_START\_WRITE

***I2C Transfer Direction Master Point of View***

I2C\_DIRECTION\_TRANSMIT  
I2C\_DIRECTION\_RECEIVE

***I2C Sequential Transfer Options***

I2C\_FIRST\_FRAME

I2C\_FIRST\_AND\_NEXT\_FRAME

I2C\_NEXT\_FRAME

I2C\_FIRST\_AND\_LAST\_FRAME

I2C\_LAST\_FRAME

## 25 HAL I2C Extension Driver

### 25.1 I2CEEx Firmware driver API description

#### 25.1.1 I2C peripheral Extended features

Comparing to other previous devices, the I2C interface for STM32F3xx devices contains the following additional features

- Possibility to disable or enable Analog Noise Filter
- Use of a configured Digital Noise Filter
- Disable or enable wakeup from Stop mode

#### 25.1.2 How to use this driver

This driver provides functions to configure Noise Filter and Wake Up Feature

1. Configure I2C Analog noise filter using the function `HAL_I2CEEx_ConfigAnalogFilter()`
2. Configure I2C Digital noise filter using the function `HAL_I2CEEx_ConfigDigitalFilter()`
3. Configure the enable or disable of I2C Wake Up Mode using the functions :
  - `HAL_I2CEEx_EnableWakeUp()`
  - `HAL_I2CEEx_DisableWakeUp()`
4. Configure the enable or disable of fast mode plus driving capability using the functions :
  - `HAL_I2CEEx_EnableFastModePlus()`
  - `HAL_I2CEEx_DisableFastModePlus()`

#### 25.1.3 Extended features functions

This section provides functions allowing to:

- Configure Noise Filters
- Configure Wake Up Feature

This section contains the following APIs:

- [`HAL\_I2CEEx\_ConfigAnalogFilter\(\)`](#)
- [`HAL\_I2CEEx\_ConfigDigitalFilter\(\)`](#)
- [`HAL\_I2CEEx\_EnableWakeUp\(\)`](#)
- [`HAL\_I2CEEx\_DisableWakeUp\(\)`](#)
- [`HAL\_I2CEEx\_EnableFastModePlus\(\)`](#)
- [`HAL\_I2CEEx\_DisableFastModePlus\(\)`](#)

#### 25.1.4 Detailed description of functions

##### `HAL_I2CEEx_ConfigAnalogFilter`

Function name      `HAL_StatusTypeDef HAL_I2CEEx_ConfigAnalogFilter(I2C_HandleTypeDef * hi2c, uint32_t AnalogFilter)`

Function description      Configure I2C Analog noise filter.

Parameters     
 

- **hi2c:** Pointer to a I2C\_HandleTypeDef structure that contains the configuration information for the specified I2Cx peripheral.
- **AnalogFilter:** New state of the Analog filter.

## Return values

- **HAL:** status

**HAL\_I2CEEx\_ConfigDigitalFilter**

## Function name

**HAL\_StatusTypeDef HAL\_I2CEEx\_ConfigDigitalFilter  
(I2C\_HandleTypeDef \* hi2c, uint32\_t DigitalFilter)**

## Function description

Configure I2C Digital noise filter.

## Parameters

- **hi2c:** Pointer to a I2C\_HandleTypeDef structure that contains the configuration information for the specified I2Cx peripheral.
- **DigitalFilter:** Coefficient of digital noise filter between Min\_Data=0x00 and Max\_Data=0x0F.

## Return values

- **HAL:** status

**HAL\_I2CEEx\_EnableWakeUp**

## Function name

**HAL\_StatusTypeDef HAL\_I2CEEx\_EnableWakeUp  
(I2C\_HandleTypeDef \* hi2c)**

## Function description

Enable I2C wakeup from stop mode.

## Parameters

- **hi2c:** Pointer to a I2C\_HandleTypeDef structure that contains the configuration information for the specified I2Cx peripheral.

## Return values

- **HAL:** status

**HAL\_I2CEEx\_DisableWakeUp**

## Function name

**HAL\_StatusTypeDef HAL\_I2CEEx\_DisableWakeUp  
(I2C\_HandleTypeDef \* hi2c)**

## Function description

Disable I2C wakeup from stop mode.

## Parameters

- **hi2c:** Pointer to a I2C\_HandleTypeDef structure that contains the configuration information for the specified I2Cx peripheral.

## Return values

- **HAL:** status

**HAL\_I2CEEx\_EnableFastModePlus**

## Function name

**void HAL\_I2CEEx\_EnableFastModePlus (uint32\_t  
ConfigFastModePlus)**

## Function description

Enable the I2C fast mode plus driving capability.

## Parameters

- **ConfigFastModePlus:** Selects the pin. This parameter can be one of the I2C Extended Fast Mode Plus values

## Return values

- **None**

## Notes

- For I2C1, fast mode plus driving capability can be enabled on all selected I2C1 pins using I2C\_FASTMODEPLUS\_I2C1 parameter or independently on each one of the following pins PB6, PB7, PB8 and PB9.
- For remaining I2C1 pins (PA14, PA15...) fast mode plus driving capability can be enabled only by using I2C\_FASTMODEPLUS\_I2C1 parameter.
- For all I2C2 pins fast mode plus driving capability can be

- enabled only by using I2C\_FASTMODEPLUS\_I2C2 parameter.
- For all I2C3 pins fast mode plus driving capability can be enabled only by using I2C\_FASTMODEPLUS\_I2C3 parameter.

### **HAL\_I2CEEx\_DisableFastModePlus**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2CEEx_DisableFastModePlus (uint32_t ConfigFastModePlus)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Disable the I2C fast mode plus driving capability.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li><b>ConfigFastModePlus:</b> Selects the pin. This parameter can be one of the I2C Extended Fast Mode Plus values</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>For I2C1, fast mode plus driving capability can be disabled on all selected I2C1 pins using I2C_FASTMODEPLUS_I2C1 parameter or independently on each one of the following pins PB6, PB7, PB8 and PB9.</li> <li>For remaining I2C1 pins (PA14, PA15...) fast mode plus driving capability can be disabled only by using I2C_FASTMODEPLUS_I2C1 parameter.</li> <li>For all I2C2 pins fast mode plus driving capability can be disabled only by using I2C_FASTMODEPLUS_I2C2 parameter.</li> <li>For all I2C3 pins fast mode plus driving capability can be disabled only by using I2C_FASTMODEPLUS_I2C3 parameter.</li> </ul> |

## **25.2 I2CEEx Firmware driver defines**

### **25.2.1 I2CEEx**

#### *I2C Extended Analog Filter*

`I2C_ANALOGFILTER_ENABLE`  
`I2C_ANALOGFILTER_DISABLE`

#### *I2C Extended Fast Mode Plus*

|                                    |                                    |
|------------------------------------|------------------------------------|
| <code>I2C_FMP_NOT_SUPPORTED</code> | Fast Mode Plus not supported       |
| <code>I2C_FASTMODEPLUS_PB6</code>  | Enable Fast Mode Plus on PB6       |
| <code>I2C_FASTMODEPLUS_PB7</code>  | Enable Fast Mode Plus on PB7       |
| <code>I2C_FASTMODEPLUS_PB8</code>  | Enable Fast Mode Plus on PB8       |
| <code>I2C_FASTMODEPLUS_PB9</code>  | Enable Fast Mode Plus on PB9       |
| <code>I2C_FASTMODEPLUS_I2C1</code> | Enable Fast Mode Plus on I2C1 pins |
| <code>I2C_FASTMODEPLUS_I2C2</code> | Enable Fast Mode Plus on I2C2 pins |
| <code>I2C_FASTMODEPLUS_I2C3</code> | Fast Mode Plus I2C3 not supported  |

## 26 HAL I2S Generic Driver

### 26.1 I2S Firmware driver registers structures

#### 26.1.1 I2S\_InitTypeDef

##### Data Fields

- *uint32\_t Mode*
- *uint32\_t Standard*
- *uint32\_t DataFormat*
- *uint32\_t MCLKOutput*
- *uint32\_t AudioFreq*
- *uint32\_t CPOL*
- *uint32\_t ClockSource*
- *uint32\_t FullDuplexMode*

##### Field Documentation

- ***uint32\_t I2S\_InitTypeDef::Mode***  
Specifies the I2S operating mode. This parameter can be a value of [\*I2S\\_Mode\*](#)
- ***uint32\_t I2S\_InitTypeDef::Standard***  
Specifies the standard used for the I2S communication. This parameter can be a value of [\*I2S\\_Standard\*](#)
- ***uint32\_t I2S\_InitTypeDef::DataFormat***  
Specifies the data format for the I2S communication. This parameter can be a value of [\*I2S\\_Data\\_Format\*](#)
- ***uint32\_t I2S\_InitTypeDef::MCLKOutput***  
Specifies whether the I2S MCLK output is enabled or not. This parameter can be a value of [\*I2S\\_MCLK\\_Output\*](#)
- ***uint32\_t I2S\_InitTypeDef::AudioFreq***  
Specifies the frequency selected for the I2S communication. This parameter can be a value of [\*I2S\\_Audio\\_Frequency\*](#)
- ***uint32\_t I2S\_InitTypeDef::CPOL***  
Specifies the idle state of the I2S clock. This parameter can be a value of [\*I2S\\_Clock\\_Polarity\*](#)
- ***uint32\_t I2S\_InitTypeDef::ClockSource***  
Specifies the I2S Clock Source. This parameter can be a value of [\*I2S\\_Clock\\_Source\*](#)
- ***uint32\_t I2S\_InitTypeDef::FullDuplexMode***  
Specifies the I2S FullDuplex mode. This parameter can be a value of [\*I2S\\_FullDuplex\\_Mode\*](#)

#### 26.1.2 I2S\_HandleTypeDef

##### Data Fields

- *SPI\_TypeDef \* Instance*
- *I2S\_InitTypeDef Init*
- *uint16\_t \* pTxBuffPtr*
- *\_IO uint16\_t TxXferSize*
- *\_IO uint16\_t TxXferCount*
- *uint16\_t \* pRxBuffPtr*
- *\_IO uint16\_t RxXferSize*
- *\_IO uint16\_t RxXferCount*

- **DMA\_HandleTypeDef \* hdmatx**
- **DMA\_HandleTypeDef \* hdmarx**
- ***\_IO HAL\_LockTypeDef Lock***
- ***\_IO HAL\_I2S\_StateTypeDef State***
- ***\_IO uint32\_t ErrorCode***

#### Field Documentation

- **SPI\_TypeDef\* I2S\_HandleTypeDef::Instance**  
I2S registers base address
- **I2S\_InitTypeDef I2S\_HandleTypeDef::Init**  
I2S communication parameters
- **uint16\_t\* I2S\_HandleTypeDef::pTxBuffPtr**  
Pointer to I2S Tx transfer buffer
- ***\_IO uint16\_t I2S\_HandleTypeDef::TxXferSize***  
I2S Tx transfer size
- ***\_IO uint16\_t I2S\_HandleTypeDef::TxXferCount***  
I2S Tx transfer Counter
- **uint16\_t\* I2S\_HandleTypeDef::pRxBuffPtr**  
Pointer to I2S Rx transfer buffer
- ***\_IO uint16\_t I2S\_HandleTypeDef::RxXferSize***  
I2S Rx transfer size
- ***\_IO uint16\_t I2S\_HandleTypeDef::RxXferCount***  
I2S Rx transfer counter (This field is initialized at the same value as transfer size at the beginning of the transfer and decremented when a sample is received.  
NbSamplesReceived = RxBufferSize-RxBufferCount)
- **DMA\_HandleTypeDef\* I2S\_HandleTypeDef::hdmatx**  
I2S Tx DMA handle parameters
- **DMA\_HandleTypeDef\* I2S\_HandleTypeDef::hdmarx**  
I2S Rx DMA handle parameters
- ***\_IO HAL\_LockTypeDef I2S\_HandleTypeDef::Lock***  
I2S locking object
- ***\_IO HAL\_I2S\_StateTypeDef I2S\_HandleTypeDef::State***  
I2S communication state
- ***\_IO uint32\_t I2S\_HandleTypeDef::ErrorCode***  
I2S Error code This parameter can be a value of [I2S\\_Error](#)

## 26.2 I2S Firmware driver API description

### 26.2.1 How to use this driver

The I2S HAL driver can be used as follows:

1. Declare a I2S\_HandleTypeDef handle structure.
2. Initialize the I2S low level resources by implement the HAL\_I2S\_MspInit() API:
  - a. Enable the SPIx interface clock.
  - b. I2S pins configuration:
    - Enable the clock for the I2S GPIOs.
    - Configure these I2S pins as alternate function pull-up.
  - c. NVIC configuration if you need to use interrupt process (HAL\_I2S\_Transmit\_IT() and HAL\_I2S\_Receive\_IT() APIs).
    - Configure the I2Sx interrupt priority.
    - Enable the NVIC I2S IRQ handle.
  - d. DMA Configuration if you need to use DMA process (HAL\_I2S\_Transmit\_DMA() and HAL\_I2S\_Receive\_DMA() APIs):
    - Declare a DMA handle structure for the Tx/Rx channel.

- Enable the DMAx interface clock.
  - Configure the declared DMA handle structure with the required Tx/Rx parameters.
  - Configure the DMA Tx/Rx Channel.
  - Associate the initialized DMA handle to the I2S DMA Tx/Rx handle.
  - Configure the priority and enable the NVIC for the transfer complete interrupt on the DMA Tx/Rx Channel.
3. Program the Mode, Standard, Data Format, MCLK Output, Audio frequency and Polarity using HAL\_I2S\_Init() function. The specific I2S interrupts (Transmission complete interrupt, RXNE interrupt and Error Interrupts) will be managed using the macros \_\_HAL\_I2S\_ENABLE\_IT() and \_\_HAL\_I2S\_DISABLE\_IT() inside the transmit and receive process. Make sure that either: I2S clock is configured based on SYSCLK or External clock source is configured after setting correctly the define constant EXTERNAL\_CLOCK\_VALUE in the stm32f3xx\_hal\_conf.h file.
4. Three mode of operations are available within this driver :

### Polling mode IO operation

- Send an amount of data in blocking mode using HAL\_I2S\_Transmit()
- Receive an amount of data in blocking mode using HAL\_I2S\_Receive()

### Interrupt mode IO operation

- Send an amount of data in non blocking mode using HAL\_I2S\_Transmit\_IT()
- At transmission end of half transfer HAL\_I2S\_TxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_TxHalfCpltCallback
- At transmission end of transfer HAL\_I2S\_TxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_TxCpltCallback
- Receive an amount of data in non blocking mode using HAL\_I2S\_Receive\_IT()
- At reception end of half transfer HAL\_I2S\_RxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_RxHalfCpltCallback
- At reception end of transfer HAL\_I2S\_RxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_RxCpltCallback
- In case of transfer Error, HAL\_I2S\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_I2S\_ErrorCallback

### DMA mode IO operation

- Send an amount of data in non blocking mode (DMA) using HAL\_I2S\_Transmit\_DMA()
- At transmission end of half transfer HAL\_I2S\_TxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_TxHalfCpltCallback
- At transmission end of transfer HAL\_I2S\_TxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_TxCpltCallback
- Receive an amount of data in non blocking mode (DMA) using HAL\_I2S\_Receive\_DMA()
- At reception end of half transfer HAL\_I2S\_RxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_RxHalfCpltCallback
- At reception end of transfer HAL\_I2S\_RxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_RxCpltCallback

- In case of transfer Error, HAL\_I2S\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_I2S\_ErrorCallback
- Pause the DMA Transfer using HAL\_I2S\_DMAPause()
- Resume the DMA Transfer using HAL\_I2S\_DMAResume()
- Stop the DMA Transfer using HAL\_I2S\_DMAStop()

### I2S HAL driver macros list

Below the list of most used macros in I2S HAL driver.

- \_\_HAL\_I2S\_ENABLE: Enable the specified SPI peripheral (in I2S mode)
- \_\_HAL\_I2S\_DISABLE: Disable the specified SPI peripheral (in I2S mode)
- \_\_HAL\_I2S\_ENABLE\_IT : Enable the specified I2S interrupts
- \_\_HAL\_I2S\_DISABLE\_IT : Disable the specified I2S interrupts
- \_\_HAL\_I2S\_GET\_FLAG: Check whether the specified I2S flag is set or not



You can refer to the I2S HAL driver header file for more useful macros

### 26.2.2 Initialization and de-initialization functions

This subsection provides a set of functions allowing to initialize and de-initialize the I2Sx peripheral in simplex mode:

- User must Implement HAL\_I2S\_MspInit() function in which he configures all related peripherals resources (CLOCK, GPIO, DMA, IT and NVIC ).
- Call the function HAL\_I2S\_Init() to configure the selected device with the selected configuration:
  - Mode
  - Standard
  - Data Format
  - MCLK Output
  - Audio frequency
  - Polarity
  - Full duplex mode
- Call the function HAL\_I2S\_DeInit() to restore the default configuration of the selected I2Sx peripheral.

This section contains the following APIs:

- [\*\*HAL\\_I2S\\_Init\(\)\*\*](#)
- [\*\*HAL\\_I2S\\_DeInit\(\)\*\*](#)
- [\*\*HAL\\_I2S\\_MspInit\(\)\*\*](#)
- [\*\*HAL\\_I2S\\_MspDeInit\(\)\*\*](#)

### 26.2.3 IO operation functions

This subsection provides a set of functions allowing to manage the I2S data transfers.

1. There are two modes of transfer:
  - Blocking mode : The communication is performed in the polling mode. The status of all data processing is returned by the same function after finishing transfer.
  - No-Blocking mode : The communication is performed using Interrupts or DMA. These functions return the status of the transfer startup. The end of the data

processing will be indicated through the dedicated I2S IRQ when using Interrupt mode or the DMA IRQ when using DMA mode.

2. Blocking mode functions are :
  - HAL\_I2S\_Transmit()
  - HAL\_I2S\_Receive()
3. No-Blocking mode functions with Interrupt are :
  - HAL\_I2S\_Transmit\_IT()
  - HAL\_I2S\_Receive\_IT()
4. No-Blocking mode functions with DMA are :
  - HAL\_I2S\_Transmit\_DMA()
  - HAL\_I2S\_Receive\_DMA()
5. A set of Transfer Complete Callbacks are provided in non Blocking mode:
  - HAL\_I2S\_TxCpltCallback()
  - HAL\_I2S\_RxCpltCallback()
  - HAL\_I2S\_ErrorCallback()

This section contains the following APIs:

- [\*HAL\\_I2S\\_Transmit\(\)\*](#)
- [\*HAL\\_I2S\\_Receive\(\)\*](#)
- [\*HAL\\_I2S\\_Transmit\\_IT\(\)\*](#)
- [\*HAL\\_I2S\\_Receive\\_IT\(\)\*](#)
- [\*HAL\\_I2S\\_Transmit\\_DMA\(\)\*](#)
- [\*HAL\\_I2S\\_Receive\\_DMA\(\)\*](#)
- [\*HAL\\_I2S\\_DMAPause\(\)\*](#)
- [\*HAL\\_I2S\\_DMAResume\(\)\*](#)
- [\*HAL\\_I2S\\_DMAStop\(\)\*](#)
- [\*HAL\\_I2S\\_IRQHandler\(\)\*](#)
- [\*HAL\\_I2S\\_TxHalfCpltCallback\(\)\*](#)
- [\*HAL\\_I2S\\_TxCpltCallback\(\)\*](#)
- [\*HAL\\_I2S\\_RxHalfCpltCallback\(\)\*](#)
- [\*HAL\\_I2S\\_RxCpltCallback\(\)\*](#)
- [\*HAL\\_I2S\\_ErrorCallback\(\)\*](#)
- [\*HAL\\_I2S\\_FullIDuplex\\_IRQHandler\(\)\*](#)
- [\*HAL\\_I2S\\_TxRxCpltCallback\(\)\*](#)
- [\*HAL\\_I2S\\_TxRxHalfCpltCallback\(\)\*](#)

#### 26.2.4 Peripheral State and Errors functions

This subsection permits to get in run-time the status of the peripheral and the data flow.

This section contains the following APIs:

- [\*HAL\\_I2S\\_GetState\(\)\*](#)
- [\*HAL\\_I2S\\_GetError\(\)\*](#)
- [\*HAL\\_I2S\\_DMAPause\(\)\*](#)
- [\*HAL\\_I2S\\_DMAResume\(\)\*](#)
- [\*HAL\\_I2S\\_DMAStop\(\)\*](#)

## 26.2.5 Detailed description of functions

### **HAL\_I2S\_Init**

|                      |                                                                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_Init (I2S_HandleTypeDef * hi2s)</b>                                                                                                                                        |
| Function description | Initializes the I2S according to the specified parameters in the I2S_InitTypeDef and create the associated handle.                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> <li>• <b>hi2s:</b> I2S handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> <li>• <b>HAL:</b> status</li> </ul>                                                                                                    |

### **HAL\_I2S\_DeInit**

|                      |                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_DeInit (I2S_HandleTypeDef * hi2s)</b>                                                                                                   |
| Function description | DeInitializes the I2S peripheral.                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                               |

### **HAL\_I2S\_MspInit**

|                      |                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2S_MspInit (I2S_HandleTypeDef * hi2s)</b>                                                                                                               |
| Function description | I2S MSP Init.                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                      |

### **HAL\_I2S\_MspDeInit**

|                      |                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2S_MspDeInit (I2S_HandleTypeDef * hi2s)</b>                                                                                                             |
| Function description | I2S MSP DeInit.                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                      |

### **HAL\_I2S\_Transmit**

|                      |                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_Transmit (I2S_HandleTypeDef * hi2s, uint16_t * pData, uint16_t Size, uint32_t Timeout)</b>                                              |
| Function description | Transmit an amount of data in blocking mode.                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |

- **pData:** a 16-bit pointer to data buffer.
  - **Size:** number of data sample to be sent:
  - **Timeout:** Timeout duration
  - **HAL:** status
- Return values**
- Notes**
- When a 16-bit data frame or a 16-bit data frame extended is selected during the I2S configuration phase, the Size parameter means the number of 16-bit data length in the transaction and when a 24-bit data frame or a 32-bit data frame is selected the Size parameter means the number of 16-bit data length.
  - The I2S is kept enabled at the end of transaction to avoid the clock de-synchronization between Master and Slave(example: audio streaming).

### HAL\_I2S\_Receive

- Function name** `HAL_StatusTypeDef HAL_I2S_Receive (I2S_HandleTypeDef * hi2s, uint16_t * pData, uint16_t Size, uint32_t Timeout)`
- Function description** Receive an amount of data in blocking mode.
- Parameters**
- **hi2s:** pointer to a I2S\_HandleTypeDef structure that contains the configuration information for I2S module
  - **pData:** a 16-bit pointer to data buffer.
  - **Size:** number of data sample to be sent:
  - **Timeout:** Timeout duration
- Return values**
- Notes**
- When a 16-bit data frame or a 16-bit data frame extended is selected during the I2S configuration phase, the Size parameter means the number of 16-bit data length in the transaction and when a 24-bit data frame or a 32-bit data frame is selected the Size parameter means the number of 16-bit data length.
  - The I2S is kept enabled at the end of transaction to avoid the clock de-synchronization between Master and Slave(example: audio streaming).
  - In I2S Master Receiver mode, just after enabling the peripheral the clock will be generate in continouse way and as the I2S is not disabled at the end of the I2S transaction.

### HAL\_I2S\_Transmit\_IT

- Function name** `HAL_StatusTypeDef HAL_I2S_Transmit_IT (I2S_HandleTypeDef * hi2s, uint16_t * pData, uint16_t Size)`
- Function description** Transmit an amount of data in non-blocking mode with Interrupt.
- Parameters**
- **hi2s:** pointer to a I2S\_HandleTypeDef structure that contains the configuration information for I2S module
  - **pData:** a 16-bit pointer to data buffer.
  - **Size:** number of data sample to be sent:
- Return values**
- **HAL:** status

- |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes | <ul style="list-style-type: none"> <li>• When a 16-bit data frame or a 16-bit data frame extended is selected during the I2S configuration phase, the <b>Size</b> parameter means the number of 16-bit data length in the transaction and when a 24-bit data frame or a 32-bit data frame is selected the <b>Size</b> parameter means the number of 16-bit data length.</li> <li>• The I2S is kept enabled at the end of transaction to avoid the clock de-synchronization between Master and Slave(example: audio streaming).</li> </ul> |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### **HAL\_I2S\_Receive\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_Receive_IT(I2S_HandleTypeDef * hi2s, uint16_t * pData, uint16_t Size)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Receive an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> <li>• <b>pData:</b> a 16-bit pointer to the Receive data buffer.</li> <li>• <b>Size:</b> number of data sample to be sent:</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                         |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• When a 16-bit data frame or a 16-bit data frame extended is selected during the I2S configuration phase, the <b>Size</b> parameter means the number of 16-bit data length in the transaction and when a 24-bit data frame or a 32-bit data frame is selected the <b>Size</b> parameter means the number of 16-bit data length.</li> <li>• The I2S is kept enabled at the end of transaction to avoid the clock de-synchronization between Master and Slave(example: audio streaming).</li> <li>• It is recommended to use DMA for the I2S receiver to avoid de-synchronisation between Master and Slave otherwise the I2S interrupt should be optimized.</li> </ul> |

### **HAL\_I2S\_IRQHandler**

|                      |                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2S_IRQHandler (I2S_HandleTypeDef * hi2s)</b>                                                                                                            |
| Function description | This function handles I2S interrupt request.                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                      |

### **HAL\_I2S\_Transmit\_DMA**

|                      |                                                                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_Transmit_DMA(I2S_HandleTypeDef * hi2s, uint16_t * pData, uint16_t Size)</b>                                                                                                                                    |
| Function description | Transmit an amount of data in non-blocking mode with DMA.                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> <li>• <b>pData:</b> a 16-bit pointer to the Transmit data buffer.</li> </ul> |

- |               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>Size:</b> number of data sample to be sent;</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Notes         | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> <li>• When a 16-bit data frame or a 16-bit data frame extended is selected during the I2S configuration phase, the Size parameter means the number of 16-bit data length in the transaction and when a 24-bit data frame or a 32-bit data frame is selected the Size parameter means the number of 16-bit data length.</li> <li>• The I2S is kept enabled at the end of transaction to avoid the clock de-synchronization between Master and Slave(example: audio streaming).</li> </ul> |

### **HAL\_I2S\_Receive\_DMA**

- |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_Receive_DMA<br/>(I2S_HandleTypeDef * hi2s, uint16_t * pData, uint16_t Size)</b>                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Receive an amount of data in non-blocking mode with DMA.                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> <li>• <b>pData:</b> a 16-bit pointer to the Receive data buffer.</li> <li>• <b>Size:</b> number of data sample to be sent;</li> </ul>                                                                                                                                                                                                                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• When a 16-bit data frame or a 16-bit data frame extended is selected during the I2S configuration phase, the Size parameter means the number of 16-bit data length in the transaction and when a 24-bit data frame or a 32-bit data frame is selected the Size parameter means the number of 16-bit data length.</li> <li>• The I2S is kept enabled at the end of transaction to avoid the clock de-synchronization between Master and Slave(example: audio streaming).</li> </ul> |

### **HAL\_I2S\_DMAPause**

- |                      |                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_DMAPause<br/>(I2S_HandleTypeDef * hi2s)</b>                                                                                             |
| Function description | Pauses the audio stream playing from the Media.                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                               |

### **HAL\_I2S\_DMAResume**

- |                      |                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_DMAResume<br/>(I2S_HandleTypeDef * hi2s)</b>                                                                                            |
| Function description | Resumes the audio stream playing from the Media.                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |

|                                   |                                                                                                                                                                    |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                     | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                               |
| <b>HAL_I2S_DMASStop</b>           |                                                                                                                                                                    |
| Function name                     | <b>HAL_StatusTypeDef HAL_I2S_DMASStop (I2S_HandleTypeDef * hi2s)</b>                                                                                               |
| Function description              | Resumes the audio stream playing from the Media.                                                                                                                   |
| Parameters                        | <ul style="list-style-type: none"> <li><b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |
| Return values                     | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                               |
| <b>HAL_I2S_TxHalfCpltCallback</b> |                                                                                                                                                                    |
| Function name                     | <b>void HAL_I2S_TxHalfCpltCallback (I2S_HandleTypeDef * hi2s)</b>                                                                                                  |
| Function description              | Tx Transfer Half completed callbacks.                                                                                                                              |
| Parameters                        | <ul style="list-style-type: none"> <li><b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |
| Return values                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                      |
| <b>HAL_I2S_TxCpltCallback</b>     |                                                                                                                                                                    |
| Function name                     | <b>void HAL_I2S_TxCpltCallback (I2S_HandleTypeDef * hi2s)</b>                                                                                                      |
| Function description              | Tx Transfer completed callbacks.                                                                                                                                   |
| Parameters                        | <ul style="list-style-type: none"> <li><b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |
| Return values                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                      |
| <b>HAL_I2S_RxHalfCpltCallback</b> |                                                                                                                                                                    |
| Function name                     | <b>void HAL_I2S_RxHalfCpltCallback (I2S_HandleTypeDef * hi2s)</b>                                                                                                  |
| Function description              | Rx Transfer half completed callbacks.                                                                                                                              |
| Parameters                        | <ul style="list-style-type: none"> <li><b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |
| Return values                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                      |
| <b>HAL_I2S_RxCpltCallback</b>     |                                                                                                                                                                    |
| Function name                     | <b>void HAL_I2S_RxCpltCallback (I2S_HandleTypeDef * hi2s)</b>                                                                                                      |
| Function description              | Rx Transfer completed callbacks.                                                                                                                                   |
| Parameters                        | <ul style="list-style-type: none"> <li><b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li> </ul> |
| Return values                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                      |

**HAL\_I2S\_ErrorCallback**

|                      |                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2S_ErrorCallback (I2S_HandleTypeDef * hi2s)</b>                                                                                                       |
| Function description | I2S error callbacks.                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                      |

**HAL\_I2S\_GetState**

|                      |                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_I2S_StateTypeDef HAL_I2S_GetState (I2S_HandleTypeDef * hi2s)</b>                                                                                            |
| Function description | Return the I2S state.                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> state</li></ul>                                                                                                |

**HAL\_I2S\_GetError**

|                      |                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_I2S_GetError (I2S_HandleTypeDef * hi2s)</b>                                                                                                        |
| Function description | Return the I2S error code.                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hi2s:</b> pointer to a I2S_HandleTypeDef structure that contains the configuration information for I2S module</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>I2S:</b> Error Code</li></ul>                                                                                           |

## 26.3 I2S Firmware driver defines

### 26.3.1 I2S

*I2S Audio Frequency*

I2S\_AUDIOFREQ\_192K  
I2S\_AUDIOFREQ\_96K  
I2S\_AUDIOFREQ\_48K  
I2S\_AUDIOFREQ\_44K  
I2S\_AUDIOFREQ\_32K  
I2S\_AUDIOFREQ\_22K  
I2S\_AUDIOFREQ\_16K  
I2S\_AUDIOFREQ\_11K  
I2S\_AUDIOFREQ\_8K  
I2S\_AUDIOFREQ\_DEFAULT  
IS\_I2S\_AUDIO\_FREQ

*I2S Clock Polarity*

I2S\_CPOL\_LOW

I2S\_CPOL\_HIGH

IS\_I2S\_CPOL

***I2S Clock Source***

I2S\_CLOCK\_EXTERNAL

I2S\_CLOCK\_SYSCLK

IS\_I2S\_CLOCKSOURCE

***I2S Data Format***

I2S\_DATAFORMAT\_16B

I2S\_DATAFORMAT\_16B\_EXTENDED

I2S\_DATAFORMAT\_24B

I2S\_DATAFORMAT\_32B

IS\_I2S\_DATA\_FORMAT

***I2S Error***

HAL\_I2S\_ERROR\_NONE      No error

HAL\_I2S\_ERROR\_TIMEOUT      Timeout error

HAL\_I2S\_ERROR\_OVR      OVR error

HAL\_I2S\_ERROR\_UDR      UDR error

HAL\_I2S\_ERROR\_DMA      DMA transfer error

HAL\_I2S\_ERROR\_UNKNOW      Unknow Error error

***I2S Exported Macros***

\_\_HAL\_I2S\_RESET\_HANDLE\_STATE      **Description:**

- Reset I2S handle state.

**Parameters:**

- \_\_HANDLE\_\_: I2S handle.

**Return value:**

- None

**Description:**

- Enable or disable the specified SPI peripheral (in I2S mode).

**Parameters:**

- \_\_HANDLE\_\_: specifies the I2S Handle.

**Return value:**

- None

\_\_HAL\_I2S\_DISABLE

\_\_HAL\_I2S\_ENABLE\_IT

**Description:**

- Enable or disable the specified I2S interrupts.

**Parameters:**

- HANDLE: specifies the I2S Handle.
- INTERRUPT: specifies the interrupt source to enable or disable. This parameter can be one of the following values:
  - I2S\_IT\_TXE: Tx buffer empty interrupt enable
  - I2S\_IT\_RXNE: RX buffer not empty interrupt enable
  - I2S\_IT\_ERR: Error interrupt enable

**Return value:**

- None

\_HAL\_I2S\_DISABLE\_IT  
\_HAL\_I2S\_GET\_IT\_SOURCE

**Description:**

- Checks if the specified I2S interrupt source is enabled or disabled.

**Parameters:**

- HANDLE: specifies the I2S Handle. This parameter can be I2S where x: 1, 2, or 3 to select the I2S peripheral.
- INTERRUPT: specifies the I2S interrupt source to check. This parameter can be one of the following values:
  - I2S\_IT\_TXE: Tx buffer empty interrupt enable
  - I2S\_IT\_RXNE: RX buffer not empty interrupt enable
  - I2S\_IT\_ERR: Error interrupt enable

**Return value:**

- The: new state of IT (TRUE or FALSE).

\_HAL\_I2S\_GET\_FLAG

**Description:**

- Checks whether the specified I2S flag is set or not.

**Parameters:**

- HANDLE: specifies the I2S Handle.
- FLAG: specifies the flag to check. This parameter can be one of the following values:
  - I2S\_FLAG\_RXNE: Receive buffer not empty flag
  - I2S\_FLAG\_TXE: Transmit buffer empty flag
  - I2S\_FLAG\_UDR: Underrun flag
  - I2S\_FLAG\_OVR: Overrun flag
  - I2S\_FLAG\_FRE: Frame error flag
  - I2S\_FLAG\_CHSIDE: Channel Side flag

- I2S\_FLAG\_BSY: Busy flag

**Return value:**

- The: new state of \_\_FLAG\_\_ (TRUE or FALSE).

`_HAL_I2S_CLEAR_OVRFLAG`

**Description:**

- Clears the I2S OVR pending flag.

**Parameters:**

- `_HANDLE_`: specifies the I2S Handle.

**Return value:**

- None

`_HAL_I2S_CLEAR_UDRFLAG`

**Description:**

- Clears the I2S UDR pending flag.

**Parameters:**

- `_HANDLE_`: specifies the I2S Handle.

**Return value:**

- None

***I2S Flag definition***

`I2S_FLAG_TXE`

`I2S_FLAG_RXNE`

`I2S_FLAG_UDR`

`I2S_FLAG_OVR`

`I2S_FLAG_FRE`

`I2S_FLAG_CHSIDE`

`I2S_FLAG_BSY`

***I2S Full Duplex Mode***

`I2S_FULLDUPLEXMODE_DISABLE`

`I2S_FULLDUPLEXMODE_ENABLE`

`IS_I2S_FULLDUPLEX_MODE`

***I2S Interrupt configuration definition***

`I2S_IT_TXE`

`I2S_IT_RXNE`

`I2S_IT_ERR`

***I2S MCLK Output***

`I2S_MCLKOUTPUT_ENABLE`

`I2S_MCLKOUTPUT_DISABLE`

`IS_I2S_MCLK_OUTPUT`

***I2S Mode***

I2S\_MODE\_SLAVE\_TX  
I2S\_MODE\_SLAVE\_RX  
I2S\_MODE\_MASTER\_TX  
I2S\_MODE\_MASTER\_RX  
IS\_I2S\_MODE

***I2S Standard***

I2S\_STANDARD\_PHILIPS  
I2S\_STANDARD\_MSB  
I2S\_STANDARD\_LSB  
I2S\_STANDARD\_PCM\_SHORT  
I2S\_STANDARD\_PCM\_LONG  
IS\_I2S\_STANDARD

## 27 HAL I2S Extension Driver

### 27.1 I2SEEx Firmware driver API description

#### 27.1.1 I2S Extended features

1. In I2S full duplex mode, each SPI peripheral is able to manage sending and receiving data simultaneously using two data lines. Each SPI peripheral has an extended block called I2Sxext ie. I2S2ext for SPI2 and I2S3ext for SPI3).
2. The Extended block is not a full SPI IP, it is used only as I2S slave to implement full duplex mode. The Extended block uses the same clock sources as its master.
3. Both I2Sx and I2Sx\_ext can be configured as transmitters or receivers. Only I2Sx can deliver SCK and WS to I2Sx\_ext in full duplex mode, where I2Sx can be I2S2 or I2S3.

#### 27.1.2 How to use this driver

Three mode of operations are available within this driver :

##### **Polling mode IO operation**

- Send and receive in the same time an amount of data in blocking mode using HAL\_I2S\_TransmitReceive()

##### **Interrupt mode IO operation**

- Send and receive in the same time an amount of data in non blocking mode using HAL\_I2S\_TransmitReceive\_IT()
- At transmission end of half transfer HAL\_I2S\_TxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_TxHalfCpltCallback
- At transmission end of transfer HAL\_I2S\_TxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_TxCpltCallback
- At reception end of half transfer HAL\_I2S\_RxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_RxHalfCpltCallback
- At reception end of transfer HAL\_I2S\_RxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_RxCpltCallback
- In case of transfer Error, HAL\_I2S\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_I2S\_ErrorCallback

##### **DMA mode IO operation**

- Send and receive an amount of data in non blocking mode (DMA) using HAL\_I2S\_TransmitReceive\_DMA()
- At transmission end of half transfer HAL\_I2S\_TxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_TxHalfCpltCallback
- At transmission end of transfer HAL\_I2S\_TxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_TxCpltCallback
- At reception end of half transfer HAL\_I2S\_RxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_RxHalfCpltCallback

- At reception end of transfer HAL\_I2S\_RxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_I2S\_RxCpltCallback
- In case of transfer Error, HAL\_I2S\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_I2S\_ErrorCallback
- Pause the DMA Transfer using HAL\_I2S\_DMAPause()
- Resume the DMA Transfer using HAL\_I2S\_DMAResume()
- Stop the DMA Transfer using HAL\_I2S\_DMAStop()

### 27.1.3 Extended features Functions

This subsection provides a set of functions allowing to manage the I2S data transfers.

1. There are two mode of transfer:
  - Blocking mode: The communication is performed in the polling mode. The status of all data processing is returned by the same function after finishing transfer.
  - No-Blocking mode: The communication is performed using Interrupts or DMA. These functions return the status of the transfer startup. The end of the data processing will be indicated through the dedicated I2S IRQ when using Interrupt mode or the DMA IRQ when using DMA mode.
2. Blocking mode functions are :
  - HAL\_I2S\_TransmitReceive()
3. No-Blocking mode functions with Interrupt are:
  - HAL\_I2S\_TransmitReceive\_IT()
  - HAL\_I2SFULLDuplex\_IRQHandler()
4. No-Blocking mode functions with DMA are:
  - HAL\_I2S\_TransmitReceive\_DMA()
5. A set of Transfer Complete Callbacks are provided in No\_Blocking mode:
  - HAL\_I2S\_TxRxCpltCallback()
  - HAL\_I2S\_TxRxHalfCpltCallback()
  - HAL\_I2S\_TxRxErrorCallback()

This section contains the following APIs:

- [\*\*\*HAL\\_I2SEEx\\_TransmitReceive\(\)\*\*\*](#)
- [\*\*\*HAL\\_I2SEEx\\_TransmitReceive\\_IT\(\)\*\*\*](#)
- [\*\*\*HAL\\_I2SEEx\\_TransmitReceive\\_DMA\(\)\*\*\*](#)

### 27.1.4 Detailed description of functions

#### HAL\_I2SEEx\_TransmitReceive

|                      |                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2SEEx_TransmitReceive<br/>(I2S_HandleTypeDef * hi2s, uint16_t * pTxData, uint16_t *<br/>pRxData, uint16_t Size, uint32_t Timeout)</b>                                                                                                                                                                    |
| Function description | Full-Duplex Transmit/Receive data in blocking mode.                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> I2S handle</li> <li>• <b>pTxData:</b> a 16-bit pointer to the Transmit data buffer.</li> <li>• <b>pRxData:</b> a 16-bit pointer to the Receive data buffer.</li> <li>• <b>Size:</b> number of data sample to be sent:</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• When a 16-bit data frame or a 16-bit data frame extended is selected during the I2S configuration phase, the <b>Size</b> parameter means the number of 16-bit data length in the</li> </ul>                                                                                               |

transaction and when a 24-bit data frame or a 32-bit data frame is selected the Size parameter means the number of 16-bit data length.

- The I2S is kept enabled at the end of transaction to avoid the clock de-synchronization between Master and Slave(example: audio streaming).

### **HAL\_I2SEEx\_TransmitReceive\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2SEEx_TransmitReceive_IT<br/>(I2S_HandleTypeDef * hi2s, uint16_t * pTxData, uint16_t * pRxData, uint16_t Size)</b>                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Full-Duplex Transmit/Receive data in non-blocking mode using Interrupt.                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li><b>hi2s:</b> I2S handle</li> <li><b>pTxData:</b> a 16-bit pointer to the Transmit data buffer.</li> <li><b>pRxData:</b> a 16-bit pointer to the Receive data buffer.</li> <li><b>Size:</b> number of data sample to be sent:</li> </ul>                                                                                                                                                                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>When a 16-bit data frame or a 16-bit data frame extended is selected during the I2S configuration phase, the Size parameter means the number of 16-bit data length in the transaction and when a 24-bit data frame or a 32-bit data frame is selected the Size parameter means the number of 16-bit data length.</li> <li>The I2S is kept enabled at the end of transaction to avoid the clock de-synchronization between Master and Slave(example: audio streaming).</li> </ul> |

### **HAL\_I2SEEx\_TransmitReceive\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2SEEx_TransmitReceive_DMA<br/>(I2S_HandleTypeDef * hi2s, uint16_t * pTxData, uint16_t * pRxData, uint16_t Size)</b>                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Full-Duplex Transmit/Receive data in non-blocking mode using DMA.                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li><b>hi2s:</b> I2S handle</li> <li><b>pTxData:</b> a 16-bit pointer to the Transmit data buffer.</li> <li><b>pRxData:</b> a 16-bit pointer to the Receive data buffer.</li> <li><b>Size:</b> number of data sample to be sent:</li> </ul>                                                                                                                                                                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>When a 16-bit data frame or a 16-bit data frame extended is selected during the I2S configuration phase, the Size parameter means the number of 16-bit data length in the transaction and when a 24-bit data frame or a 32-bit data frame is selected the Size parameter means the number of 16-bit data length.</li> <li>The I2S is kept enabled at the end of transaction to avoid the clock de-synchronization between Master and Slave(example: audio streaming).</li> </ul> |

**HAL\_I2S\_FullDuplex\_IRQHandler**

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2S_FullDuplex_IRQHandler (I2S_HandleTypeDef * hi2s)</b>      |
| Function description | This function handles I2S/I2Sext interrupt requests in full-duplex mode.  |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hi2s:</b> I2S handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>      |

**HAL\_I2S\_TxRxCpltCallback**

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2S_TxRxCpltCallback (I2S_HandleTypeDef * hi2s)</b>           |
| Function description | Tx and Rx Transfer completed callbacks.                                   |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hi2s:</b> I2S handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>             |

**HAL\_I2S\_TxRxHalfCpltCallback**

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>void HAL_I2S_TxRxHalfCpltCallback (I2S_HandleTypeDef * hi2s)</b>       |
| Function description | Tx and Rx Transfer half completed callbacks.                              |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hi2s:</b> I2S handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>             |

**HAL\_I2S\_DMAPause**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_DMAPause (I2S_HandleTypeDef * hi2s)</b>        |
| Function description | Pauses the audio stream playing from the Media.                             |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hi2s:</b> : I2S handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>               |

**HAL\_I2S\_DMAResume**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_DMAResume (I2S_HandleTypeDef * hi2s)</b>       |
| Function description | Resumes the audio stream playing from the Media.                            |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hi2s:</b> : I2S handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>               |

**HAL\_I2S\_DMAStop**

|                      |                                                                     |
|----------------------|---------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_I2S_DMAStop (I2S_HandleTypeDef * hi2s)</b> |
| Function description | Resumes the audio stream playing from the Media.                    |

---

|               |                                                                             |
|---------------|-----------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hi2s:</b> I2S handle</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

## 27.2 I2SEEx Firmware driver defines

### 27.2.1 I2SEEx

#### *I2S Extended Exported Macros*

##### I2SxEXT

`_HAL_I2SEXT_ENABLE`

##### Description:

- Enable or disable the specified I2SExt peripheral.

##### Parameters:

- `_HANDLE_`: specifies the I2S Handle.

##### Return value:

- None

`_HAL_I2SEXT_DISABLE`

`_HAL_I2SEXT_ENABLE_IT`

##### Description:

- Enable or disable the specified I2SExt interrupts.

##### Parameters:

- `_HANDLE_`: specifies the I2S Handle.
- `_INTERRUPT_`: specifies the interrupt source to enable or disable. This parameter can be one of the following values:
  - `I2S_IT_TXE`: Tx buffer empty interrupt enable
  - `I2S_IT_RXNE`: RX buffer not empty interrupt enable
  - `I2S_IT_ERR`: Error interrupt enable

##### Return value:

- None

`_HAL_I2SEXT_DISABLE_IT`

`_HAL_I2SEXT_GET_IT_SOURCE`

##### Description:

- Checks if the specified I2SExt interrupt source is enabled or disabled.

##### Parameters:

- `_HANDLE_`: specifies the I2S Handle. This parameter can be I2S where x: 1, 2, or 3 to select the I2S peripheral.
- `_INTERRUPT_`: specifies the I2S interrupt source to check. This parameter can be one of the following values:
  - `I2S_IT_TXE`: Tx buffer empty interrupt

- enable
  - I2S\_IT\_RXNE: RX buffer not empty interrupt enable
  - I2S\_IT\_ERR: Error interrupt enable

**Return value:**

- The: new state of \_\_IT\_\_ (TRUE or FALSE).

[\\_\\_HAL\\_I2SEXT\\_GET\\_FLAG](#)

**Description:**

- Checks whether the specified I2SExt flag is set or not.

**Parameters:**

- \_\_HANDLE\_\_: specifies the I2S Handle.
- \_\_FLAG\_\_: specifies the flag to check. This parameter can be one of the following values:
  - I2S\_FLAG\_RXNE: Receive buffer not empty flag
  - I2S\_FLAG\_TXE: Transmit buffer empty flag
  - I2S\_FLAG\_UDR: Underrun flag
  - I2S\_FLAG\_OVR: Overrun flag
  - I2S\_FLAG\_FRE: Frame error flag
  - I2S\_FLAG\_CHSIDE: Channel Side flag
  - I2S\_FLAG\_BSY: Busy flag

**Return value:**

- The: new state of \_\_FLAG\_\_ (TRUE or FALSE).

[\\_\\_HAL\\_I2SEXT\\_CLEAR\\_OVRFLAG](#)

**Description:**

- Clears the I2SExt OVR pending flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the I2S Handle.

**Return value:**

- None

[\\_\\_HAL\\_I2SEXT\\_CLEAR\\_UDRFLAG](#)

**Description:**

- Clears the I2SExt UDR pending flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the I2S Handle.

**Return value:**

- None

## 28 HAL IRDA Generic Driver

### 28.1 IRDA Firmware driver registers structures

#### 28.1.1 IRDA\_InitTypeDef

##### Data Fields

- *uint32\_t BaudRate*
- *uint32\_t WordLength*
- *uint32\_t Parity*
- *uint32\_t Mode*
- *uint8\_t Prescaler*
- *uint16\_t PowerMode*

##### Field Documentation

- ***uint32\_t IRDA\_InitTypeDef::BaudRate***

This member configures the IRDA communication baud rate. The baud rate register is computed using the following formula: Baud Rate Register = ((PCLKx) / ((hirda->Init.BaudRate)))

- ***uint32\_t IRDA\_InitTypeDef::WordLength***

Specifies the number of data bits transmitted or received in a frame. This parameter can be a value of [IRDAEx\\_Word\\_Length](#)

- ***uint32\_t IRDA\_InitTypeDef::Parity***

Specifies the parity mode. This parameter can be a value of [IRDA\\_Parity](#)  
**Note:**When parity is enabled, the computed parity is inserted at the MSB position of the transmitted data (9th bit when the word length is set to 9 data bits; 8th bit when the word length is set to 8 data bits).

- ***uint32\_t IRDA\_InitTypeDef::Mode***

Specifies whether the Receive or Transmit mode is enabled or disabled. This parameter can be a value of [IRDA\\_Transfer\\_Mode](#)

- ***uint8\_t IRDA\_InitTypeDef::Prescaler***

Specifies the Prescaler value for dividing the UART/USART source clock to achieve low-power frequency.

**Note:**Prescaler value 0 is forbidden

- ***uint16\_t IRDA\_InitTypeDef::PowerMode***

Specifies the IRDA power mode. This parameter can be a value of [IRDA\\_Low\\_Power](#)

#### 28.1.2 IRDA\_HandleTypeDef

##### Data Fields

- *USART\_TypeDef \* Instance*
- *IRDA\_InitTypeDef Init*
- *uint8\_t \* pTxBuffPtr*
- *uint16\_t TxXferSize*
- *\_\_IO uint16\_t TxXferCount*
- *uint8\_t \* pRxBuffPtr*
- *uint16\_t RxXferSize*
- *\_\_IO uint16\_t RxXferCount*
- *uint16\_t Mask*
- *DMA\_HandleTypeDef \* hdmatx*
- *DMA\_HandleTypeDef \* hdmarx*

- ***HAL\_LockTypeDef Lock***
- ***\_IO HAL\_IRDA\_StateTypeDef gState***
- ***\_IO HAL\_IRDA\_StateTypeDef RxState***
- ***\_IO uint32\_t ErrorCode***

#### Field Documentation

- ***USART\_TypeDef\* IRDA\_HandleTypeDef::Instance***  
IRDA registers base address
- ***IRDA\_InitTypeDef IRDA\_HandleTypeDef::Init***  
IRDA communication parameters
- ***uint8\_t\* IRDA\_HandleTypeDef::pTxBuffPtr***  
Pointer to IRDA Tx transfer Buffer
- ***uint16\_t IRDA\_HandleTypeDef::TxXferSize***  
IRDA Tx Transfer size
- ***\_IO uint16\_t IRDA\_HandleTypeDef::TxXferCount***  
IRDA Tx Transfer Counter
- ***uint8\_t\* IRDA\_HandleTypeDef::pRxBuffPtr***  
Pointer to IRDA Rx transfer Buffer
- ***uint16\_t IRDA\_HandleTypeDef::RxXferSize***  
IRDA Rx Transfer size
- ***\_IO uint16\_t IRDA\_HandleTypeDef::RxXferCount***  
IRDA Rx Transfer Counter
- ***uint16\_t IRDA\_HandleTypeDef::Mask***  
IRDA RX RDR register mask
- ***DMA\_HandleTypeDef\* IRDA\_HandleTypeDef::hdmatx***  
IRDA Tx DMA Handle parameters
- ***DMA\_HandleTypeDef\* IRDA\_HandleTypeDef::hdmarx***  
IRDA Rx DMA Handle parameters
- ***HAL\_LockTypeDef IRDA\_HandleTypeDef::Lock***  
Locking object
- ***\_IO HAL\_IRDA\_StateTypeDef IRDA\_HandleTypeDef::gState***  
IRDA state information related to global Handle management and also related to Tx operations. This parameter can be a value of **HAL\_IRDA\_StateTypeDef**
- ***\_IO HAL\_IRDA\_StateTypeDef IRDA\_HandleTypeDef::RxState***  
IRDA state information related to Rx operations. This parameter can be a value of **HAL\_IRDA\_StateTypeDef**
- ***\_IO uint32\_t IRDA\_HandleTypeDef::ErrorCode***  
IRDA Error code This parameter can be a value of **IRDA\_Error**

## 28.2 IRDA Firmware driver API description

### 28.2.1 How to use this driver

The IRDA HAL driver can be used as follows:

1. Declare a IRDA\_HandleTypeDef handle structure (eg. IRDA\_HandleTypeDef hirda).
2. Initialize the IRDA low level resources by implementing the HAL\_IRDA\_MspInit() API in setting the associated USART or UART in IRDA mode:
  - Enable the USARTx/UARTx interface clock.
  - USARTx/UARTx pins configuration:
    - Enable the clock for the USARTx/UARTx GPIOs.
    - Configure these USARTx/UARTx pins (TX as alternate function pull-up, RX as alternate function Input).

- NVIC configuration if you need to use interrupt process (HAL\_IRDA\_Transmit\_IT() and HAL\_IRDA\_Receive\_IT() APIs):
    - Configure the USARTx/UARTx interrupt priority.
    - Enable the NVIC USARTx/UARTx IRQ handle.
    - The specific IRDA interrupts (Transmission complete interrupt, RXNE interrupt and Error Interrupts) will be managed using the macros \_\_HAL\_IRDA\_ENABLE\_IT() and \_\_HAL\_IRDA\_DISABLE\_IT() inside the transmit and receive process.
  - DMA Configuration if you need to use DMA process (HAL\_IRDA\_Transmit\_DMA() and HAL\_IRDA\_Receive\_DMA() APIs):
    - Declare a DMA handle structure for the Tx/Rx channel.
    - Enable the DMAx interface clock.
    - Configure the declared DMA handle structure with the required Tx/Rx parameters.
    - Configure the DMA Tx/Rx channel.
    - Associate the initialized DMA handle to the IRDA DMA Tx/Rx handle.
    - Configure the priority and enable the NVIC for the transfer complete interrupt on the DMA Tx/Rx channel.
3. Program the Baud Rate, Word Length and Parity and Mode(Receiver/Transmitter), the normal or low power mode and the clock prescaler in the hirda handle Init structure.
  4. Initialize the IRDA registers by calling the HAL\_IRDA\_Init() API:
    - This API configures also the low level Hardware (GPIO, CLOCK, CORTEX...etc) by calling the customized HAL\_IRDA\_MspInit() API. The specific IRDA interrupts (Transmission complete interrupt, RXNE interrupt and Error Interrupts) will be managed using the macros \_\_HAL\_IRDA\_ENABLE\_IT() and \_\_HAL\_IRDA\_DISABLE\_IT() inside the transmit and receive process.
  5. Three operation modes are available within this driver :

### **Polling mode IO operation**

- Send an amount of data in blocking mode using HAL\_IRDA\_Transmit()
- Receive an amount of data in blocking mode using HAL\_IRDA\_Receive()

### **Interrupt mode IO operation**

- Send an amount of data in non-blocking mode using HAL\_IRDA\_Transmit\_IT()
- At transmission end of transfer HAL\_IRDA\_TxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_IRDA\_TxCpltCallback()
- Receive an amount of data in non-blocking mode using HAL\_IRDA\_Receive\_IT()
- At reception end of transfer HAL\_IRDA\_RxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_IRDA\_RxCpltCallback()
- In case of transfer Error, HAL\_IRDA\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_IRDA\_ErrorCallback()

### **DMA mode IO operation**

- Send an amount of data in non-blocking mode (DMA) using HAL\_IRDA\_Transmit\_DMA()
- At transmission half of transfer HAL\_IRDA\_TxHalfCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_IRDA\_TxHalfCpltCallback()
- At transmission end of transfer HAL\_IRDA\_TxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_IRDA\_TxCpltCallback()
- Receive an amount of data in non-blocking mode (DMA) using HAL\_IRDA\_Receive\_DMA()

- At reception half of transfer HAL\_IRDA\_RxHalfCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_IRDA\_RxHalfCpltCallback()
- At reception end of transfer HAL\_IRDA\_RxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_IRDA\_RxCpltCallback()
- In case of transfer Error, HAL\_IRDA\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_IRDA\_ErrorCallback()

### IRDA HAL driver macros list

Below the list of most used macros in IRDA HAL driver.

- \_\_HAL\_IRDA\_ENABLE: Enable the IRDA peripheral
- \_\_HAL\_IRDA\_DISABLE: Disable the IRDA peripheral
- \_\_HAL\_IRDA\_GET\_FLAG : Check whether the specified IRDA flag is set or not
- \_\_HAL\_IRDA\_CLEAR\_FLAG : Clear the specified IRDA pending flag
- \_\_HAL\_IRDA\_ENABLE\_IT: Enable the specified IRDA interrupt
- \_\_HAL\_IRDA\_DISABLE\_IT: Disable the specified IRDA interrupt
- \_\_HAL\_IRDA\_GET\_IT\_SOURCE: Check whether or not the specified IRDA interrupt is enabled



You can refer to the IRDA HAL driver header file for more useful macros

### 28.2.2 Initialization and Configuration functions

This subsection provides a set of functions allowing to initialize the USARTx in asynchronous IRDA mode.

- For the asynchronous mode only these parameters can be configured:
  - Baud Rate
  - Word Length
  - Parity
  - Power mode
  - Prescaler setting
  - Receiver/transmitter modes

The HAL\_IRDA\_Init() API follows the USART asynchronous configuration procedures (details for the procedures are available in reference manual).

This section contains the following APIs:

- [\*\*HAL\\_IRDA\\_Init\(\)\*\*](#)
- [\*\*HAL\\_IRDA\\_DeInit\(\)\*\*](#)
- [\*\*HAL\\_IRDA\\_MspInit\(\)\*\*](#)
- [\*\*HAL\\_IRDA\\_MspDeInit\(\)\*\*](#)

### 28.2.3 IO operation functions

This subsection provides a set of functions allowing to manage the IRDA data transfers.

IrDA is a half duplex communication protocol. If the Transmitter is busy, any data on the IrDA receive line will be ignored by the IrDA decoder and if the Receiver is busy, data on the TX from the USART to IrDA will not be encoded by IrDA. While receiving data, transmission should be avoided as the data to be transmitted could be corrupted.

1. There are two mode of transfer:
  - Blocking mode: the communication is performed in polling mode. The HAL status of all data processing is returned by the same function after finishing transfer.
  - Non-Blocking mode: the communication is performed using Interrupts or DMA, these API's return the HAL status. The end of the data processing will be indicated through the dedicated IRDA IRQ when using Interrupt mode or the DMA IRQ when using DMA mode. The HAL\_IRDA\_TxCpltCallback(), HAL\_IRDA\_RxCpltCallback() user callbacks will be executed respectively at the end of the Transmit or Receive process. The HAL\_IRDA\_ErrorCallback() user callback will be executed when a communication error is detected
2. Blocking mode APIs are :
  - HAL\_IRDA\_Transmit()
  - HAL\_IRDA\_Receive()
3. Non Blocking mode APIs with Interrupt are :
  - HAL\_IRDA\_Transmit\_IT()
  - HAL\_IRDA\_Receive\_IT()
  - HAL\_IRDA\_IRQHandler()
4. Non Blocking mode functions with DMA are :
  - HAL\_IRDA\_Transmit\_DMA()
  - HAL\_IRDA\_Receive\_DMA()
  - HAL\_IRDA\_DMAPause()
  - HAL\_IRDA\_DMAResume()
  - HAL\_IRDA\_DMAStop()
5. A set of Transfer Complete Callbacks are provided in Non Blocking mode:
  - HAL\_IRDA\_TxHalfCpltCallback()
  - HAL\_IRDA\_TxCpltCallback()
  - HAL\_IRDA\_RxHalfCpltCallback()
  - HAL\_IRDA\_RxCpltCallback()
  - HAL\_IRDA\_ErrorCallback()
6. Non-Blocking mode transfers could be aborted using Abort API's :
  - HAL\_IRDA\_Abort()
  - HAL\_IRDA\_AbortTransmit()
  - HAL\_IRDA\_AbortReceive()
  - HAL\_IRDA\_Abort\_IT()
  - HAL\_IRDA\_AbortTransmit\_IT()
  - HAL\_IRDA\_AbortReceive\_IT()
7. For Abort services based on interrupts (HAL\_IRDA\_Abortxxx\_IT), a set of Abort Complete Callbacks are provided:
  - HAL\_IRDA\_AbortCpltCallback()
  - HAL\_IRDA\_AbortTransmitCpltCallback()
  - HAL\_IRDA\_AbortReceiveCpltCallback()
8. In Non-Blocking mode transfers, possible errors are split into 2 categories. Errors are handled as follows :
  - Error is considered as Recoverable and non blocking : Transfer could go till end, but error severity is to be evaluated by user : this concerns Frame Error, Parity Error or Noise Error in Interrupt mode reception . Received character is then retrieved and stored in Rx buffer, Error code is set to allow user to identify error type, and HAL\_IRDA\_ErrorCallback() user callback is executed. Transfer is kept ongoing on IRDA side. If user wants to abort it, Abort services should be called by user.
  - Error is considered as Blocking : Transfer could not be completed properly and is aborted. This concerns Overrun Error In Interrupt mode reception and all errors in DMA mode. Error code is set to allow user to identify error type, and HAL\_IRDA\_ErrorCallback() user callback is executed.

This section contains the following APIs:

- [\*HAL\\_IRDA\\_Transmit\(\)\*](#)
- [\*HAL\\_IRDA\\_Receive\(\)\*](#)
- [\*HAL\\_IRDA\\_Transmit\\_IT\(\)\*](#)
- [\*HAL\\_IRDA\\_Receive\\_IT\(\)\*](#)
- [\*HAL\\_IRDA\\_Transmit\\_DMA\(\)\*](#)
- [\*HAL\\_IRDA\\_Receive\\_DMA\(\)\*](#)
- [\*HAL\\_IRDA\\_DMAPause\(\)\*](#)
- [\*HAL\\_IRDA\\_DMAResume\(\)\*](#)
- [\*HAL\\_IRDA\\_DMAStop\(\)\*](#)
- [\*HAL\\_IRDA\\_Abort\(\)\*](#)
- [\*HAL\\_IRDA\\_AbortTransmit\(\)\*](#)
- [\*HAL\\_IRDA\\_AbortReceive\(\)\*](#)
- [\*HAL\\_IRDA\\_Abort\\_IT\(\)\*](#)
- [\*HAL\\_IRDA\\_AbortTransmit\\_IT\(\)\*](#)
- [\*HAL\\_IRDA\\_AbortReceive\\_IT\(\)\*](#)
- [\*HAL\\_IRDA\\_IRQHandler\(\)\*](#)
- [\*HAL\\_IRDA\\_TxCpltCallback\(\)\*](#)
- [\*HAL\\_IRDA\\_TxHalfCpltCallback\(\)\*](#)
- [\*HAL\\_IRDA\\_RxCpltCallback\(\)\*](#)
- [\*HAL\\_IRDA\\_RxHalfCpltCallback\(\)\*](#)
- [\*HAL\\_IRDA\\_ErrorCallback\(\)\*](#)
- [\*HAL\\_IRDA\\_AbortCpltCallback\(\)\*](#)
- [\*HAL\\_IRDA\\_AbortTransmitCpltCallback\(\)\*](#)
- [\*HAL\\_IRDA\\_AbortReceiveCpltCallback\(\)\*](#)

#### 28.2.4 Peripheral State and Error functions

This subsection provides a set of functions allowing to return the State of IrDA communication process and also return Peripheral Errors occurred during communication process

- [\*HAL\\_IRDA\\_GetState\(\)\*](#) API can be helpful to check in run-time the state of the IRDA peripheral handle.
- [\*HAL\\_IRDA\\_GetError\(\)\*](#) checks in run-time errors that could occur during communication.

This section contains the following APIs:

- [\*HAL\\_IRDA\\_GetState\(\)\*](#)
- [\*HAL\\_IRDA\\_GetError\(\)\*](#)

#### 28.2.5 Detailed description of functions

##### **HAL\_IRDA\_Init**

|                      |                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_IRDA_Init (IRDA_HandleTypeDef * hirda)</code>                                                                                                            |
| Function description | Initialize the IRDA mode according to the specified parameters in the IRDA_InitTypeDef and initialize the associated handle.                                                         |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li></ul> |

|               |                                                                        |
|---------------|------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul> |
|---------------|------------------------------------------------------------------------|

### **HAL\_IRDA\_DelInit**

|               |                                                                        |
|---------------|------------------------------------------------------------------------|
| Function name | <b>HAL_StatusTypeDef HAL_IRDA_DelInit (IRDA_HandleTypeDef * hirda)</b> |
|---------------|------------------------------------------------------------------------|

Function description Delinitialize the IRDA peripheral.

|            |                                                                                                                                                                                        |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul> |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|               |                                                                        |
|---------------|------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul> |
|---------------|------------------------------------------------------------------------|

### **HAL\_IRDA\_MspInit**

|               |                                                           |
|---------------|-----------------------------------------------------------|
| Function name | <b>void HAL_IRDA_MspInit (IRDA_HandleTypeDef * hirda)</b> |
|---------------|-----------------------------------------------------------|

Function description Initialize the IRDA MSP.

|            |                                                                                                                                                                                        |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul> |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|               |                                                                 |
|---------------|-----------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |
|---------------|-----------------------------------------------------------------|

### **HAL\_IRDA\_MspDelInit**

|               |                                                              |
|---------------|--------------------------------------------------------------|
| Function name | <b>void HAL_IRDA_MspDelInit (IRDA_HandleTypeDef * hirda)</b> |
|---------------|--------------------------------------------------------------|

Function description Delinitialize the IRDA MSP.

|            |                                                                                                                                                                                        |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul> |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|               |                                                                 |
|---------------|-----------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |
|---------------|-----------------------------------------------------------------|

### **HAL\_IRDA\_Transmit**

|               |                                                                                                                           |
|---------------|---------------------------------------------------------------------------------------------------------------------------|
| Function name | <b>HAL_StatusTypeDef HAL_IRDA_Transmit (IRDA_HandleTypeDef * hirda, uint8_t * pData, uint16_t Size, uint32_t Timeout)</b> |
|---------------|---------------------------------------------------------------------------------------------------------------------------|

Function description Send an amount of data in blocking mode.

|            |                                                                                                                                                                                                                                                                                                                                              |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> <li>• <b>pData:</b> Pointer to data buffer.</li> <li>• <b>Size:</b> Amount of data to be sent.</li> <li>• <b>Timeout:</b> Specify timeout value.</li> </ul> |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|               |                                                                        |
|---------------|------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul> |
|---------------|------------------------------------------------------------------------|

**HAL\_IRDA\_Receive**

|                      |                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IRDA_Receive</b><br><b>(IRDA_HandleTypeDef * hirda, uint8_t * pData, uint16_t Size, uint32_t Timeout)</b>                                                                                                                                                                                                               |
| Function description | Receive an amount of data in blocking mode.                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> <li>• <b>pData:</b> Pointer to data buffer.</li> <li>• <b>Size:</b> Amount of data to be received.</li> <li>• <b>Timeout:</b> Specify timeout value.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                           |

**HAL\_IRDA\_Transmit\_IT**

|                      |                                                                                                                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IRDA_Transmit_IT</b><br><b>(IRDA_HandleTypeDef * hirda, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                                       |
| Function description | Send an amount of data in interrupt mode.                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> <li>• <b>pData:</b> Pointer to data buffer.</li> <li>• <b>Size:</b> Amount of data to be sent.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                     |

**HAL\_IRDA\_Receive\_IT**

|                      |                                                                                                                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IRDA_Receive_IT</b><br><b>(IRDA_HandleTypeDef * hirda, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                                            |
| Function description | Receive an amount of data in interrupt mode.                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> <li>• <b>pData:</b> Pointer to data buffer.</li> <li>• <b>Size:</b> Amount of data to be received.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                         |

**HAL\_IRDA\_Transmit\_DMA**

|                      |                                                                                                                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IRDA_Transmit_DMA</b><br><b>(IRDA_HandleTypeDef * hirda, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                                      |
| Function description | Send an amount of data in DMA mode.                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be sent.</li> </ul> |

|                             |                                                                                                                                                                                                                                                                                          |
|-----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values               | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                     |
| <b>HAL_IRDA_Receive_DMA</b> |                                                                                                                                                                                                                                                                                          |
| Function name               | <b>HAL_StatusTypeDef HAL_IRDA_Receive_DMA<br/>(IRDA_HandleTypeDef * hirda, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                                           |
| Function description        | Receive an amount of data in DMA mode.                                                                                                                                                                                                                                                   |
| Parameters                  | <ul style="list-style-type: none"> <li><b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> <li><b>pData:</b> Pointer to data buffer.</li> <li><b>Size:</b> Amount of data to be received.</li> </ul> |
| Return values               | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                     |
| Notes                       | <ul style="list-style-type: none"> <li>When the IRDA parity is enabled (PCE = 1), the received data contains the parity bit (MSB position).</li> </ul>                                                                                                                                   |
| <b>HAL_IRDA_DMAPause</b>    |                                                                                                                                                                                                                                                                                          |
| Function name               | <b>HAL_StatusTypeDef HAL_IRDA_DMAPause<br/>(IRDA_HandleTypeDef * hirda)</b>                                                                                                                                                                                                              |
| Function description        | Pause the DMA Transfer.                                                                                                                                                                                                                                                                  |
| Parameters                  | <ul style="list-style-type: none"> <li><b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul>                                                                                                     |
| Return values               | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                     |
| <b>HAL_IRDA_DMAResume</b>   |                                                                                                                                                                                                                                                                                          |
| Function name               | <b>HAL_StatusTypeDef HAL_IRDA_DMAResume<br/>(IRDA_HandleTypeDef * hirda)</b>                                                                                                                                                                                                             |
| Function description        | Resume the DMA Transfer.                                                                                                                                                                                                                                                                 |
| Parameters                  | <ul style="list-style-type: none"> <li><b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified UART module.</li> </ul>                                                                                                     |
| Return values               | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                     |
| <b>HAL_IRDA_DMAStop</b>     |                                                                                                                                                                                                                                                                                          |
| Function name               | <b>HAL_StatusTypeDef HAL_IRDA_DMAStop<br/>(IRDA_HandleTypeDef * hirda)</b>                                                                                                                                                                                                               |
| Function description        | Stop the DMA Transfer.                                                                                                                                                                                                                                                                   |
| Parameters                  | <ul style="list-style-type: none"> <li><b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified UART module.</li> </ul>                                                                                                     |
| Return values               | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                     |

**HAL\_IRDA\_Abort**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IRDA_Abort (IRDA_HandleTypeDef * hirda)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Abort ongoing transfers (blocking mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified UART module.</li> </ul>                                                                                                                                                                                                                                                                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable IRDA Interrupts (Tx and Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort (in case of transfer in DMA mode)Set handle State to READY</li> <li>• This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.</li> </ul> |

**HAL\_IRDA\_AbortTransmit**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IRDA_AbortTransmit (IRDA_HandleTypeDef * hirda)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Abort ongoing Transmit transfer (blocking mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified UART module.</li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Tx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable IRDA Interrupts (Tx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort (in case of transfer in DMA mode)Set handle State to READY</li> <li>• This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.</li> </ul> |

**HAL\_IRDA\_AbortReceive**

|                      |                                                                                                                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IRDA_AbortReceive (IRDA_HandleTypeDef * hirda)</b>                                                                                                                                                                                                                |
| Function description | Abort ongoing Receive transfer (blocking mode).                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified UART module.</li> </ul>                                                                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Rx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable IRDA Interrupts (Rx)Disable the DMA transfer in the peripheral register (if</li> </ul> |

- enabled)Abort DMA transfer by calling HAL\_DMA\_Abort (in case of transfer in DMA mode)Set handle State to READY
- This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.

### **HAL\_IRDA\_Abort\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IRDA_Abort_IT<br/>(IRDA_HandleTypeDef * hirda)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Abort ongoing transfers (Interrupt mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li><b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified UART module.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>This procedure could be used for aborting any ongoing transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable IRDA Interrupts (Tx and Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort_IT (in case of transfer in DMA mode)Set handle State to READYAt abort completion, call user abort complete callback</li> <li>This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).</li> </ul> |

### **HAL\_IRDA\_AbortTransmit\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IRDA_AbortTransmit_IT<br/>(IRDA_HandleTypeDef * hirda)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Abort ongoing Transmit transfer (Interrupt mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified UART module.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Notes                | <ul style="list-style-type: none"> <li>This procedure could be used for aborting any ongoing Tx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable IRDA Interrupts (Tx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort_IT (in case of transfer in DMA mode)Set handle State to READYAt abort completion, call user abort complete callback</li> <li>This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).</li> </ul> |

**HAL\_IRDA\_AbortReceive\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IRDA_AbortReceive_IT<br/>(IRDA_HandleTypeDef * hirda)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Abort ongoing Receive transfer (Interrupt mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified UART module.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Rx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable IRDA Interrupts (Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort_IT (in case of transfer in DMA mode)Set handle State to READYAt abort completion, call user abort complete callback</li> <li>• This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).</li> </ul> |

**HAL\_IRDA\_IRQHandler**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_IRDA_IRQHandler (IRDA_HandleTypeDef * hirda)</b>                                                                                                                           |
| Function description | Handle IRDA interrupt request.                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                        |

**HAL\_IRDA\_TxCpltCallback**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_IRDA_TxCpltCallback (IRDA_HandleTypeDef * hirda)</b>                                                                                                                       |
| Function description | Tx Transfer completed callback.                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                        |

**HAL\_IRDA\_RxCpltCallback**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_IRDA_RxCpltCallback (IRDA_HandleTypeDef * hirda)</b>                                                                                                                       |
| Function description | Rx Transfer completed callback.                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul> |

|               |                                                                 |
|---------------|-----------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |
|---------------|-----------------------------------------------------------------|

### **HAL\_IRDA\_TxHalfCpltCallback**

Function name      **void HAL\_IRDA\_TxHalfCpltCallback (IRDA\_HandleTypeDef \* hirda)**

Function description    Tx Half Transfer completed callback.

Parameters            • **hirda:** Pointer to a IRDA\_HandleTypeDef structure that contains the configuration information for the specified USART module.

|               |                                                                 |
|---------------|-----------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |
|---------------|-----------------------------------------------------------------|

### **HAL\_IRDA\_RxHalfCpltCallback**

Function name      **void HAL\_IRDA\_RxHalfCpltCallback (IRDA\_HandleTypeDef \* hirda)**

Function description    Rx Half Transfer complete callback.

Parameters            • **hirda:** Pointer to a IRDA\_HandleTypeDef structure that contains the configuration information for the specified IRDA module.

|               |                                                                 |
|---------------|-----------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |
|---------------|-----------------------------------------------------------------|

### **HAL\_IRDA\_ErrorCallback**

Function name      **void HAL\_IRDA\_ErrorCallback (IRDA\_HandleTypeDef \* hirda)**

Function description    IRDA error callback.

Parameters            • **hirda:** Pointer to a IRDA\_HandleTypeDef structure that contains the configuration information for the specified IRDA module.

|               |                                                                 |
|---------------|-----------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |
|---------------|-----------------------------------------------------------------|

### **HAL\_IRDA\_AbortCpltCallback**

Function name      **void HAL\_IRDA\_AbortCpltCallback (IRDA\_HandleTypeDef \* hirda)**

Function description    IRDA Abort Complete callback.

Parameters            • **hirda:** Pointer to a IRDA\_HandleTypeDef structure that contains the configuration information for the specified IRDA module.

|               |                                                                 |
|---------------|-----------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |
|---------------|-----------------------------------------------------------------|

### **HAL\_IRDA\_AbortTransmitCpltCallback**

Function name      **void HAL\_IRDA\_AbortTransmitCpltCallback (IRDA\_HandleTypeDef \* hirda)**

Function description    IRDA Abort Complete callback.

|               |                                                                                                                                                                                        |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                        |

**HAL\_IRDA\_AbortReceiveCpltCallback**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_IRDA_AbortReceiveCpltCallback<br/>(IRDA_HandleTypeDef * hirda)</b>                                                                                                         |
| Function description | IRDA Abort Receive Complete callback.                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                        |

**HAL\_IRDA\_GetState**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_IRDA_StateTypeDef HAL_IRDA_GetState<br/>(IRDA_HandleTypeDef * hirda)</b>                                                                                                        |
| Function description | Return the IRDA handle state.                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                                  |

**HAL\_IRDA\_GetError**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_IRDA_GetError (IRDA_HandleTypeDef * hirda)</b>                                                                                                                         |
| Function description | Return the IRDA handle error code.                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hirda:</b> Pointer to a IRDA_HandleTypeDef structure that contains the configuration information for the specified IRDA module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>IRDA:</b> Error Code</li> </ul>                                                                                                            |

## 28.3 IRDA Firmware driver defines

### 28.3.1 IRDA

***IRDA DMA Rx***

|                            |                      |
|----------------------------|----------------------|
| <b>IRDA_DMA_RX_DISABLE</b> | IRDA DMA RX disabled |
| <b>IRDA_DMA_RX_ENABLE</b>  | IRDA DMA RX enabled  |

***IRDA DMA Tx***

|                            |                      |
|----------------------------|----------------------|
| <b>IRDA_DMA_TX_DISABLE</b> | IRDA DMA TX disabled |
| <b>IRDA_DMA_TX_ENABLE</b>  | IRDA DMA TX enabled  |

***IRDA Error***

|                     |                    |
|---------------------|--------------------|
| HAL_IRDA_ERROR_NONE | No error           |
| HAL_IRDA_ERROR_PE   | Parity error       |
| HAL_IRDA_ERROR_NE   | Noise error        |
| HAL_IRDA_ERROR_FE   | frame error        |
| HAL_IRDA_ERROR_ORE  | Overrun error      |
| HAL_IRDA_ERROR_DMA  | DMA transfer error |
| HAL_IRDA_ERROR_BUSY | Busy Error         |

***IRDA Exported Macros***

|                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>_HAL_IRDA_RESET_HANDLE_STA<br/>TE</code> | <b>Description:</b><br><ul style="list-style-type: none"><li>Reset IRDA handle state.</li></ul> <b>Parameters:</b><br><ul style="list-style-type: none"><li><code>_HANDLE_</code>: IRDA handle.</li></ul> <b>Return value:</b><br><ul style="list-style-type: none"><li>None</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <code>_HAL_IRDA_FLUSH_DRREGISTER</code>        | <b>Description:</b><br><ul style="list-style-type: none"><li>Flush the IRDA DR register.</li></ul> <b>Parameters:</b><br><ul style="list-style-type: none"><li><code>_HANDLE_</code>: specifies the IRDA Handle.</li></ul> <b>Return value:</b><br><ul style="list-style-type: none"><li>None</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <code>_HAL_IRDA_CLEAR_FLAG</code>              | <b>Description:</b><br><ul style="list-style-type: none"><li>Clear the specified IRDA pending flag.</li></ul> <b>Parameters:</b><br><ul style="list-style-type: none"><li><code>_HANDLE_</code>: specifies the IRDA Handle.</li><li><code>_FLAG_</code>: specifies the flag to check. This parameter can be any combination of the following values:<ul style="list-style-type: none"><li>- <code>IRDA_CLEAR_PEF</code></li><li>- <code>IRDA_CLEAR_FEF</code></li><li>- <code>IRDA_CLEAR_NEF</code></li><li>- <code>IRDA_CLEAR_OREF</code></li><li>- <code>IRDA_CLEAR_TCF</code></li><li>- <code>IRDA_CLEAR_IDLEF</code></li></ul></li></ul> <b>Return value:</b><br><ul style="list-style-type: none"><li>None</li></ul> |
| <code>_HAL_IRDA_CLEAR_PEFLAG</code>            | <b>Description:</b><br><ul style="list-style-type: none"><li>Clear the IRDA PE pending flag.</li></ul> <b>Parameters:</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

- `__HANDLE__`: specifies the IRDA Handle.

**Return value:**

- None

`__HAL_IRDA_CLEAR_FEFLAG`

**Description:**

- Clear the IRDA FE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the IRDA Handle.

**Return value:**

- None

`__HAL_IRDA_CLEAR_NEFLAG`

**Description:**

- Clear the IRDA NE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the IRDA Handle.

**Return value:**

- None

`__HAL_IRDA_CLEAR_OREFLAG`

**Description:**

- Clear the IRDA ORE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the IRDA Handle.

**Return value:**

- None

`__HAL_IRDA_CLEAR_IDLEFLAG`

**Description:**

- Clear the IRDA IDLE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the IRDA Handle.

**Return value:**

- None

`__HAL_IRDA_GET_FLAG`

**Description:**

- Check whether the specified IRDA flag is set or not.

**Parameters:**

- `__HANDLE__`: specifies the IRDA Handle.
- `__FLAG__`: specifies the flag to check. This parameter can be one of the following values:
  - `IRDA_FLAG_RXACK` Receive enable acknowledge flag
  - `IRDA_FLAG_TXACK` Transmit enable acknowledge flag

- IRDA\_FLAG\_BUSY Busy flag
- IRDA\_FLAG\_ABRF Auto Baud rate detection flag
- IRDA\_FLAG\_ABRE Auto Baud rate detection error flag
- IRDA\_FLAG\_TXE Transmit data register empty flag
- IRDA\_FLAG\_TC Transmission Complete flag
- IRDA\_FLAG\_RXNE Receive data register not empty flag
- IRDA\_FLAG\_ORE OverRun Error flag
- IRDA\_FLAG\_NE Noise Error flag
- IRDA\_FLAG\_FE Framing Error flag
- IRDA\_FLAG\_PE Parity Error flag

**Return value:**

- The new state of \_\_FLAG\_\_ (TRUE or FALSE).

[\\_\\_HAL\\_IRDA\\_ENABLE\\_IT](#)**Description:**

- Enable the specified IRDA interrupt.

**Parameters:**

- \_\_HANDLE\_\_: specifies the IRDA Handle.
- \_\_INTERRUPT\_\_: specifies the IRDA interrupt source to enable. This parameter can be one of the following values:
  - IRDA\_IT\_TXE Transmit Data Register empty interrupt
  - IRDA\_IT\_TC Transmission complete interrupt
  - IRDA\_IT\_RXNE Receive Data register not empty interrupt
  - IRDA\_IT\_IDLE Idle line detection interrupt
  - IRDA\_IT\_PE Parity Error interrupt
  - IRDA\_IT\_ERR Error interrupt(Frame error, noise error, overrun error)

**Return value:**

- None

[\\_\\_HAL\\_IRDA\\_DISABLE\\_IT](#)**Description:**

- Disable the specified IRDA interrupt.

**Parameters:**

- \_\_HANDLE\_\_: specifies the IRDA Handle.
- \_\_INTERRUPT\_\_: specifies the IRDA interrupt source to disable. This parameter can be one of the following values:
  - IRDA\_IT\_TXE Transmit Data Register empty interrupt
  - IRDA\_IT\_TC Transmission complete

- interrupt
  - IRDA\_IT\_RXNE Receive Data register not empty interrupt
  - IRDA\_IT\_IDLE Idle line detection interrupt
  - IRDA\_IT\_PE Parity Error interrupt
  - IRDA\_IT\_ERR Error interrupt(Frame error, noise error, overrun error)

**Return value:**

- None

\_\_HAL\_IRDA\_GET\_IT

**Description:**

- Check whether the specified IRDA interrupt has occurred or not.

**Parameters:**

- \_\_HANDLE\_\_: specifies the IRDA Handle.
- \_\_IT\_\_: specifies the IRDA interrupt source to check. This parameter can be one of the following values:
  - IRDA\_IT\_TXE Transmit Data Register empty interrupt
  - IRDA\_IT\_TC Transmission complete interrupt
  - IRDA\_IT\_RXNE Receive Data register not empty interrupt
  - IRDA\_IT\_IDLE Idle line detection interrupt
  - IRDA\_IT\_ORE OverRun Error interrupt
  - IRDA\_IT\_NE Noise Error interrupt
  - IRDA\_IT\_FE Framing Error interrupt
  - IRDA\_IT\_PE Parity Error interrupt

**Return value:**

- The: new state of \_\_IT\_\_ (TRUE or FALSE).

\_\_HAL\_IRDA\_GET\_IT\_SOURCE

**Description:**

- Check whether the specified IRDA interrupt source is enabled or not.

**Parameters:**

- \_\_HANDLE\_\_: specifies the IRDA Handle.
- \_\_IT\_\_: specifies the IRDA interrupt source to check. This parameter can be one of the following values:
  - IRDA\_IT\_TXE Transmit Data Register empty interrupt
  - IRDA\_IT\_TC Transmission complete interrupt
  - IRDA\_IT\_RXNE Receive Data register not empty interrupt
  - IRDA\_IT\_IDLE Idle line detection interrupt

- IRDA\_IT\_ERR Framing, overrun or noise error interrupt
- IRDA\_IT\_PE Parity Error interrupt

**Return value:**

- The new state of \_\_IT\_\_ (TRUE or FALSE).

**\_HAL\_IRDA\_CLEAR\_IT****Description:**

- Clear the specified IRDA ISR flag, in setting the proper ICR register flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the IRDA Handle.
- \_\_IT\_CLEAR\_\_: specifies the interrupt clear register flag that needs to be set to clear the corresponding interrupt. This parameter can be one of the following values:
  - IRDA\_CLEAR\_PEF Parity Error Clear Flag
  - IRDA\_CLEAR\_FEF Framing Error Clear Flag
  - IRDA\_CLEAR\_NEF Noise detected Clear Flag
  - IRDA\_CLEAR\_OREF OverRun Error Clear Flag
  - IRDA\_CLEAR\_TCF Transmission Complete Clear Flag

**Return value:**

- None

**\_HAL\_IRDA\_SEND\_REQ****Description:**

- Set a specific IRDA request flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the IRDA Handle.
- \_\_REQ\_\_: specifies the request flag to set. This parameter can be one of the following values:
  - IRDA\_AUTOBAUD\_REQUEST Auto-Baud Rate Request
  - IRDA\_RXDATA\_FLUSH\_REQUEST Receive Data flush Request
  - IRDA\_TXDATA\_FLUSH\_REQUEST Transmit data flush Request

**Return value:**

- None

**\_HAL\_IRDA\_ONE\_BIT\_SAMPLE\_ENABLE****Description:**

- Enable the IRDA one bit sample method.

**Parameters:**

- `_HANDLE_`: specifies the IRDA Handle.

**Return value:**

- None

`_HAL_IRDA_ONE_BIT_SAMPLE_DISABLE`

**Description:**

- Disable the IRDA one bit sample method.

**Parameters:**

- `_HANDLE_`: specifies the IRDA Handle.

**Return value:**

- None

`_HAL_IRDA_ENABLE`

**Description:**

- Enable UART/USART associated to IRDA Handle.

**Parameters:**

- `_HANDLE_`: specifies the IRDA Handle.

**Return value:**

- None

`_HAL_IRDA_DISABLE`

**Description:**

- Disable UART/USART associated to IRDA Handle.

**Parameters:**

- `_HANDLE_`: specifies the IRDA Handle.

**Return value:**

- None

***IRDA Flags***

|                              |                                       |
|------------------------------|---------------------------------------|
| <code>IRDA_FLAG_RXACK</code> | IRDA Receive enable acknowledge flag  |
| <code>IRDA_FLAG_TEACK</code> | IRDA Transmit enable acknowledge flag |
| <code>IRDA_FLAG_BUSY</code>  | IRDA Busy flag                        |
| <code>IRDA_FLAG_ABRF</code>  | IRDA Auto baud rate flag              |
| <code>IRDA_FLAG_ABRE</code>  | IRDA Auto baud rate error             |
| <code>IRDA_FLAG_TXE</code>   | IRDA Transmit data register empty     |
| <code>IRDA_FLAG_TC</code>    | IRDA Transmission complete            |
| <code>IRDA_FLAG_RXNE</code>  | IRDA Read data register not empty     |
| <code>IRDA_FLAG_ORE</code>   | IRDA Overrun error                    |
| <code>IRDA_FLAG_NE</code>    | IRDA Noise error                      |
| <code>IRDA_FLAG_FE</code>    | IRDA Framing error                    |
| <code>IRDA_FLAG_PE</code>    | IRDA Parity error                     |

***IRDA interruptions flags mask***

**IRDA\_IT\_MASK**      IRDA Interruptions flags mask

***IRDA Interrupts Definition***

|              |                                                |
|--------------|------------------------------------------------|
| IRDA_IT_PE   | IRDA Parity error interruption                 |
| IRDA_IT_TXE  | IRDA Transmit data register empty interruption |
| IRDA_IT_TC   | IRDA Transmission complete interruption        |
| IRDA_IT_RXNE | IRDA Read data register not empty interruption |
| IRDA_IT_IDLE | IRDA Idle interruption                         |
| IRDA_IT_ERR  | IRDA Error interruption                        |
| IRDA_IT_ORE  | IRDA Overrun error interruption                |
| IRDA_IT_NE   | IRDA Noise error interruption                  |
| IRDA_IT_FE   | IRDA Frame error interruption                  |

***IRDA Interruption Clear Flags***

|                  |                                  |
|------------------|----------------------------------|
| IRDA_CLEAR_PEF   | Parity Error Clear Flag          |
| IRDA_CLEAR_FEF   | Framing Error Clear Flag         |
| IRDA_CLEAR_NEF   | Noise detected Clear Flag        |
| IRDA_CLEAR_OREF  | OverRun Error Clear Flag         |
| IRDA_CLEAR_IDLEF | IDLE line detected Clear Flag    |
| IRDA_CLEAR_TCF   | Transmission Complete Clear Flag |

***IRDA Low Power***

|                         |                        |
|-------------------------|------------------------|
| IRDA_POWERMODE_NORMAL   | IRDA normal power mode |
| IRDA_POWERMODE_LOWPOWER | IRDA low power mode    |

***IRDA Mode***

|                   |                                       |
|-------------------|---------------------------------------|
| IRDA_MODE_DISABLE | Associated UART disabled in IRDA mode |
| IRDA_MODE_ENABLE  | Associated UART enabled in IRDA mode  |

***IRDA One Bit Sampling***

|                             |                           |
|-----------------------------|---------------------------|
| IRDA_ONE_BIT_SAMPLE_DISABLE | One-bit sampling disabled |
| IRDA_ONE_BIT_SAMPLE_ENABLE  | One-bit sampling enabled  |

***IRDA Parity***

|                  |             |
|------------------|-------------|
| IRDA_PARITY_NONE | No parity   |
| IRDA_PARITY EVEN | Even parity |
| IRDA_PARITY ODD  | Odd parity  |

***IRDA Request Parameters***

|                           |                             |
|---------------------------|-----------------------------|
| IRDA_AUTOBAUD_REQUEST     | Auto-Baud Rate Request      |
| IRDA_RXDATA_FLUSH_REQUEST | Receive Data flush Request  |
| IRDA_TXDATA_FLUSH_REQUEST | Transmit data flush Request |

***IRDA State***

`IRDA_STATE_DISABLE` IRDA disabled

`IRDA_STATE_ENABLE` IRDA enabled

***IRDA Transfer Mode***

`IRDA_MODE_RX` RX mode

`IRDA_MODE_TX` TX mode

`IRDA_MODE_TX_RX` RX and TX mode

## 29 HAL IRDA Extension Driver

### 29.1 IRDAEx Firmware driver defines

#### 29.1.1 IRDAEx

##### *IRDA Word Length*

IRDA\_WORDLENGTH\_8B 8-bit long frame

IRDA\_WORDLENGTH\_9B 9-bit long frame

## 30 HAL IWDG Generic Driver

### 30.1 IWDG Firmware driver registers structures

#### 30.1.1 IWDG\_InitTypeDef

##### Data Fields

- *uint32\_t Prescaler*
- *uint32\_t Reload*
- *uint32\_t Window*

##### Field Documentation

- ***IWDG\_InitTypeDef::Prescaler***  
Select the prescaler of the IWDG. This parameter can be a value of [\*IWDG\\_Prescaler\*](#)
- ***IWDG\_InitTypeDef::Reload***  
Specifies the IWDG down-counter reload value. This parameter must be a number between Min\_Data = 0 and Max\_Data = 0x0FFFU
- ***IWDG\_InitTypeDef::Window***  
Specifies the window value to be compared to the down-counter. This parameter must be a number between Min\_Data = 0 and Max\_Data = 0x0FFFU

#### 30.1.2 IWDG\_HandleTypeDef

##### Data Fields

- *IWDG\_TypeDef \* Instance*
- *IWDG\_InitTypeDef Init*

##### Field Documentation

- ***IWDG\_HandleTypeDef::Instance***  
Register base address
- ***IWDG\_HandleTypeDef::Init***  
IWDG required parameters

### 30.2 IWDG Firmware driver API description

#### 30.2.1 IWDG Generic features

- The IWDG can be started by either software or hardware (configurable through option byte).
- The IWDG is clocked by Low-Speed clock (LSI) and thus stays active even if the main clock fails.
- Once the IWDG is started, the LSI is forced ON and both can not be disabled. The counter starts counting down from the reset value (0xFFFFU). When it reaches the end of count value (0x000U) a reset signal is generated (IWDG reset).
- Whenever the key value 0x0000 AAAA is written in the IWDG\_KR register, the IWDG\_RLR value is reloaded in the counter and the watchdog reset is prevented.
- The IWDG is implemented in the VDD voltage domain that is still functional in STOP and STANDBY mode (IWDG reset can wake-up from STANDBY). IWDGRST flag in RCC\_CSR register can be used to inform when an IWDG reset occurs.
- Debug mode : When the microcontroller enters debug mode (core halted), the IWDG counter either continues to work normally or stops, depending on DBG\_IWDG\_STOP

configuration bit in DBG module, accessible through  
`__HAL_DBGMCU_FREEZE_IWDG()` and `__HAL_DBGMCU_UNFREEZE_IWDG()`  
 macros

Min-max timeout value @41KHz (LSI): ~0.1ms / ~25.5s The IWDG timeout may vary due to LSI frequency dispersion. STM32F3xx devices provide the capability to measure the LSI frequency (LSI clock connected internally to TIM16 CH1 input capture). The measured value can be used to have an IWDG timeout with an acceptable accuracy.

### 30.2.2 How to use this driver

1. Use IWDG using `HAL_IWDG_Init()` function to :
  - Enable instance by writing Start keyword in IWDG\_KEY register. LSI clock is forced ON and IWDG counter starts downcounting.
  - Enable write access to configuration register: IWDG\_PR, IWDG\_RLR & IWDG\_WINR.
  - Configure the IWDG prescaler and counter reload value. This reload value will be loaded in the IWDG counter each time the watchdog is reloaded, then the IWDG will start counting down from this value.
  - wait for status flags to be reset"
  - Depending on window parameter:
    - If Window Init parameter is same as Window register value, nothing more is done but reload counter value in order to exit function with exact time base.
    - Else modify Window register. This will automatically reload watchdog counter.
2. Then the application program must refresh the IWDG counter at regular intervals during normal operation to prevent an MCU reset, using `HAL_IWDG_Refresh()` function.

#### IWDG HAL driver macros list

Below the list of most used macros in IWDG HAL driver:

- `__HAL_IWDG_START`: Enable the IWDG peripheral
- `__HAL_IWDG_RELOAD_COUNTER`: Reloads IWDG counter with value defined in the reload register

### 30.2.3 Initialization and Start functions

This section provides functions allowing to:

- Initialize the IWDG according to the specified parameters in the IWDG\_InitTypeDef of associated handle.
- Manage Window option.
- Once initialization is performed in `HAL_IWDG_Init` function, Watchdog is reloaded in order to exit function with correct time base.

This section contains the following APIs:

- [`HAL\_IWDG\_Init\(\)`](#)

### 30.2.4 IO operation functions

This section provides functions allowing to:

- Refresh the IWDG.

This section contains the following APIs:

- [`HAL\_IWDG\_Refresh\(\)`](#)

### 30.2.5 Detailed description of functions

#### **HAL\_IWDG\_Init**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IWDG_Init (IWDG_HandleTypeDef * hiwdg)</b>                                                                                                                    |
| Function description | Initialize the IWDG according to the specified parameters in the IWDG_InitTypeDef and start watchdog.                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hiwdg:</b> pointer to a IWDG_HandleTypeDef structure that contains the configuration information for the specified IWDG module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                 |

#### **HAL\_IWDG\_Refresh**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_IWDG_Refresh (IWDG_HandleTypeDef * hiwdg)</b>                                                                                                                 |
| Function description | Refresh the IWDG.                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hiwdg:</b> pointer to a IWDG_HandleTypeDef structure that contains the configuration information for the specified IWDG module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                 |

## 30.3 IWDG Firmware driver defines

### 30.3.1 IWDG

#### *IWDG Exported Macros*

##### \_HAL\_IWDG\_START

##### **Description:**

- Enable the IWDG peripheral.

##### **Parameters:**

- \_HANDLE\_: IWDG handle

##### **Return value:**

- None

##### \_HAL\_IWDG\_RELOAD\_COUNTER

##### **Description:**

- Reload IWDG counter with value defined in the reload register (write access to IWDG\_PR, IWDG\_RLR & IWDG\_WINR registers disabled).

##### **Parameters:**

- \_HANDLE\_: IWDG handle

##### **Return value:**

- None

***IWDG Prescaler***

|                    |                            |
|--------------------|----------------------------|
| IWDG_PRESCALER_4   | IWDG prescaler set to 4    |
| IWDG_PRESCALER_8   | IWDG prescaler set to 8    |
| IWDG_PRESCALER_16  | IWDG prescaler set to 16   |
| IWDG_PRESCALER_32  | IWDG prescaler set to 32   |
| IWDG_PRESCALER_64  | IWDG prescaler set to 64   |
| IWDG_PRESCALER_128 | IWDG prescaler set to 128U |
| IWDG_PRESCALER_256 | IWDG prescaler set to 256U |

***IWDG Window option***

IWDG\_WINDOW\_DISABLE

## 31 HAL NAND Generic Driver

### 31.1 NAND Firmware driver registers structures

#### 31.1.1 NAND\_IDTypeDef

##### Data Fields

- *uint8\_t Maker\_Id*
- *uint8\_t Device\_Id*
- *uint8\_t Third\_Id*
- *uint8\_t Fourth\_Id*

##### Field Documentation

- *uint8\_t NAND\_IDTypeDef::Maker\_Id*
- *uint8\_t NAND\_IDTypeDef::Device\_Id*
- *uint8\_t NAND\_IDTypeDef::Third\_Id*
- *uint8\_t NAND\_IDTypeDef::Fourth\_Id*

#### 31.1.2 NAND\_AddressTypeDef

##### Data Fields

- *uint16\_t Page*
- *uint16\_t Plane*
- *uint16\_t Block*

##### Field Documentation

- *uint16\_t NAND\_AddressTypeDef::Page*  
NAND memory Page address
- *uint16\_t NAND\_AddressTypeDef::Plane*  
NAND memory Plane address
- *uint16\_t NAND\_AddressTypeDef::Block*  
NAND memory Block address

#### 31.1.3 NAND\_DeviceConfigTypeDef

##### Data Fields

- *uint32\_t PageSize*
- *uint32\_t SpareAreaSize*
- *uint32\_t BlockSize*
- *uint32\_t BlockNbr*
- *uint32\_t PlaneNbr*
- *uint32\_t PlaneSize*
- *FunctionalState ExtraCommandEnable*

##### Field Documentation

- *uint32\_t NAND\_DeviceConfigTypeDef::PageSize*  
NAND memory page (without spare area) size measured in bytes for 8 bits addressing or words for 16 bits addressing
- *uint32\_t NAND\_DeviceConfigTypeDef::SpareAreaSize*  
NAND memory spare area size measured in bytes for 8 bits addressing or words for 16 bits addressing

- ***uint32\_t NAND\_DeviceConfigTypeDef::BlockSize***  
NAND memory block size measured in number of pages
- ***uint32\_t NAND\_DeviceConfigTypeDef::BlockNbr***  
NAND memory number of total blocks
- ***uint32\_t NAND\_DeviceConfigTypeDef::PlaneNbr***  
NAND memory number of planes
- ***uint32\_t NAND\_DeviceConfigTypeDef::PlaneSize***  
NAND memory plane size measured in number of blocks
- ***FunctionalState NAND\_DeviceConfigTypeDef::ExtraCommandEnable***  
NAND extra command needed for Page reading mode. This parameter is mandatory for some NAND parts after the read command (NAND\_CMD\_AREA\_TRUE1) and before DATA reading sequence. Example: Toshiba TTH58BYG3S0HBAI6. This parameter could be ENABLE or DISABLE Please check the Read Mode sequenece in the NAND device datasheet

### 31.1.4 NAND\_HandleTypeDef

#### Data Fields

- ***FMC\_NAND\_TypeDef \* Instance***
- ***FMC\_NAND\_InitTypeDef Init***
- ***HAL\_LockTypeDef Lock***
- ***\_\_IO HAL\_NAND\_StateTypeDef State***
- ***NAND\_DeviceConfigTypeDef Config***

#### Field Documentation

- ***FMC\_NAND\_TypeDef\* NAND\_HandleTypeDef::Instance***  
Register base address
- ***FMC\_NAND\_InitTypeDef NAND\_HandleTypeDef::Init***  
NAND device control configuration parameters
- ***HAL\_LockTypeDef NAND\_HandleTypeDef::Lock***  
NAND locking object
- ***\_\_IO HAL\_NAND\_StateTypeDef NAND\_HandleTypeDef::State***  
NAND device access state
- ***NAND\_DeviceConfigTypeDef NAND\_HandleTypeDef::Config***  
NAND phusical characteristic information structure

## 31.2 NAND Firmware driver API description

### 31.2.1 How to use this driver

This driver is a generic layered driver which contains a set of APIs used to control NAND flash memories. It uses the FMC layer functions to interface with NAND devices. This driver is used as follows:

- NAND flash memory configuration sequence using the function HAL\_NAND\_Init() with control and timing parameters for both common and attribute spaces.
- Read NAND flash memory maker and device IDs using the function HAL\_NAND\_Read\_ID(). The read information is stored in the NAND\_ID\_TypeDef structure declared by the function caller.
- Access NAND flash memory by read/write operations using the functions HAL\_NAND\_Read\_Page\_8b()/HAL\_NAND\_Read\_SpareArea\_8b(),  
HAL\_NAND\_Write\_Page\_8b()/HAL\_NAND\_Write\_SpareArea\_8b(),  
HAL\_NAND\_Read\_Page\_16b()/HAL\_NAND\_Read\_SpareArea\_16b(),  
HAL\_NAND\_Write\_Page\_16b()/HAL\_NAND\_Write\_SpareArea\_16b() to read/write page(s)/spare area(s). These functions use specific device information (Block, page

size..) predefined by the user in the HAL\_NAND\_Info\_TypeDef structure. The read/write address information is contained by the Nand\_Address\_TypeDef structure passed as parameter.

- Perform NAND flash Reset chip operation using the function HAL\_NAND\_Reset().
- Perform NAND flash erase block operation using the function HAL\_NAND\_Erase\_Block(). The erase block address information is contained in the Nand\_Address\_TypeDef structure passed as parameter.
- Read the NAND flash status operation using the function HAL\_NAND\_Read\_Status().
- You can also control the NAND device by calling the control APIs HAL\_NAND\_ECC\_Enable() / HAL\_NAND\_ECC\_Disable() to respectively enable/disable the ECC code correction feature or the function HAL\_NAND\_GetECC() to get the ECC correction code.
- You can monitor the NAND device HAL state by calling the function HAL\_NAND\_GetState()



This driver is a set of generic APIs which handle standard NAND flash operations. If a NAND flash device contains different operations and/or implementations, it should be implemented separately.

### 31.2.2 NAND Initialization and de-initialization functions

This section provides functions allowing to initialize/de-initialize the NAND memory

This section contains the following APIs:

- [\*HAL\\_NAND\\_Init\(\)\*](#)
- [\*HAL\\_NAND\\_DelInit\(\)\*](#)
- [\*HAL\\_NAND\\_MspInit\(\)\*](#)
- [\*HAL\\_NAND\\_MspDelInit\(\)\*](#)
- [\*HAL\\_NAND\\_IRQHandler\(\)\*](#)
- [\*HAL\\_NAND\\_ITCallback\(\)\*](#)
- [\*HAL\\_NAND\\_ConfigDevice\(\)\*](#)
- [\*HAL\\_NAND\\_Read\\_ID\(\)\*](#)

### 31.2.3 NAND Input and Output functions

This section provides functions allowing to use and control the NAND memory

This section contains the following APIs:

- [\*HAL\\_NAND\\_Read\\_ID\(\)\*](#)
- [\*HAL\\_NAND\\_Reset\(\)\*](#)
- [\*HAL\\_NAND\\_ConfigDevice\(\)\*](#)
- [\*HAL\\_NAND\\_Read\\_Page\\_8b\(\)\*](#)
- [\*HAL\\_NAND\\_Read\\_Page\\_16b\(\)\*](#)
- [\*HAL\\_NAND\\_Write\\_Page\\_8b\(\)\*](#)
- [\*HAL\\_NAND\\_Write\\_Page\\_16b\(\)\*](#)
- [\*HAL\\_NAND\\_Read\\_SpareArea\\_8b\(\)\*](#)
- [\*HAL\\_NAND\\_Read\\_SpareArea\\_16b\(\)\*](#)
- [\*HAL\\_NAND\\_Write\\_SpareArea\\_8b\(\)\*](#)
- [\*HAL\\_NAND\\_Write\\_SpareArea\\_16b\(\)\*](#)
- [\*HAL\\_NAND\\_Erase\\_Block\(\)\*](#)
- [\*HAL\\_NAND\\_Read\\_Status\(\)\*](#)
- [\*HAL\\_NAND\\_Address\\_Inc\(\)\*](#)

### 31.2.4 NAND Control functions

This subsection provides a set of functions allowing to control dynamically the NAND interface.

This section contains the following APIs:

- [\*HAL\\_NAND\\_ECC\\_Enable\(\)\*](#)
- [\*HAL\\_NAND\\_ECC\\_Disable\(\)\*](#)
- [\*HAL\\_NAND\\_GetECC\(\)\*](#)

### 31.2.5 NAND State functions

This subsection permits to get in run-time the status of the NAND controller and the data flow.

This section contains the following APIs:

- [\*HAL\\_NAND\\_GetState\(\)\*](#)

### 31.2.6 Detailed description of functions

#### **HAL\_NAND\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_NAND_Init (NAND_HandleTypeDef * hnand, FMC_NAND_PCC_TimingTypeDef * ComSpace_Timing, FMC_NAND_PCC_TimingTypeDef * AttSpace_Timing)</code>                                                                                                                                                               |
| Function description | Perform NAND memory Initialization sequence.                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnand</b>: pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li>• <b>ComSpace_Timing</b>: pointer to Common space timing structure</li> <li>• <b>AttSpace_Timing</b>: pointer to Attribute space timing structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL</b>: status</li> </ul>                                                                                                                                                                                                                                                              |

#### **HAL\_NAND\_DelInit**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_NAND_DelInit (NAND_HandleTypeDef * hnand)</code>                                                                                             |
| Function description | Perform NAND memory De-Initialization sequence.                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnand</b>: pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL</b>: status</li> </ul>                                                                                                   |

#### **HAL\_NAND\_ConfigDevice**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_NAND_ConfigDevice (NAND_HandleTypeDef * hnand, NAND_DeviceConfigTypeDef * pDeviceConfig)</code>                                              |
| Function description | Configure the device: Enter the physical parameters of the device.                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnand</b>: pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> </ul> |

- **pDeviceConfig:** pointer to NAND\_DeviceConfigTypeDef structure
- Return values      • **HAL:** status

### **HAL\_NAND\_Read\_ID**

- Function name      **HAL\_StatusTypeDef HAL\_NAND\_Read\_ID  
(NAND\_HandleTypeDef \* hñand, NAND\_IDTypeDef \*  
pNAND\_ID)**
- Function description      Read the NAND memory electronic signature.
- Parameters
  - **hñand:** pointer to a NAND\_HandleTypeDef structure that contains the configuration information for NAND module.
  - **pNAND\_ID:** NAND ID structure
- Return values      • **HAL:** status

### **HAL\_NAND\_MspInit**

- Function name      **void HAL\_NAND\_MspInit (NAND\_HandleTypeDef \* hñand)**
- Function description      NAND MSP Init.
- Parameters
  - **hñand:** pointer to a NAND\_HandleTypeDef structure that contains the configuration information for NAND module.
- Return values      • **None**

### **HAL\_NAND\_MspDelInit**

- Function name      **void HAL\_NAND\_MspDelInit (NAND\_HandleTypeDef \* hñand)**
- Function description      NAND MSP Delinit.
- Parameters
  - **hñand:** pointer to a NAND\_HandleTypeDef structure that contains the configuration information for NAND module.
- Return values      • **None**

### **HAL\_NAND\_IRQHandler**

- Function name      **void HAL\_NAND\_IRQHandler (NAND\_HandleTypeDef \* hñand)**
- Function description      This function handles NAND device interrupt request.
- Parameters
  - **hñand:** pointer to a NAND\_HandleTypeDef structure that contains the configuration information for NAND module.
- Return values      • **HAL:** status

### **HAL\_NAND\_ITCallback**

- Function name      **void HAL\_NAND\_ITCallback (NAND\_HandleTypeDef \* hñand)**
- Function description      NAND interrupt feature callback.
- Parameters
  - **hñand:** pointer to a NAND\_HandleTypeDef structure that contains the configuration information for NAND module.

|                                   |                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                     | <ul style="list-style-type: none"> <li>None</li> </ul>                                                                                                                                                                                                                                                                                                           |
| <b>HAL_NAND_Reset</b>             |                                                                                                                                                                                                                                                                                                                                                                  |
| Function name                     | <b>HAL_StatusTypeDef HAL_NAND_Reset<br/>(NAND_HandleTypeDef * hndl)</b>                                                                                                                                                                                                                                                                                          |
| Function description              | NAND memory reset.                                                                                                                                                                                                                                                                                                                                               |
| Parameters                        | <ul style="list-style-type: none"> <li><b>hndl</b>: pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> </ul>                                                                                                                                                                                            |
| Return values                     | <ul style="list-style-type: none"> <li><b>HAL</b>: status</li> </ul>                                                                                                                                                                                                                                                                                             |
| <b>HAL_NAND_Read_Page_8b</b>      |                                                                                                                                                                                                                                                                                                                                                                  |
| Function name                     | <b>HAL_StatusTypeDef HAL_NAND_Read_Page_8b<br/>(NAND_HandleTypeDef * hndl, NAND_AddressTypeDef * pAddress, uint8_t * pBuffer, uint32_t NumPageToRead)</b>                                                                                                                                                                                                        |
| Function description              | Read Page(s) from NAND memory block (8-bits addressing)                                                                                                                                                                                                                                                                                                          |
| Parameters                        | <ul style="list-style-type: none"> <li><b>hndl</b>: pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li><b>pAddress</b>: pointer to NAND address structure</li> <li><b>pBuffer</b>: pointer to destination read buffer</li> <li><b>NumPageToRead</b>: number of pages to read from block</li> </ul>  |
| Return values                     | <ul style="list-style-type: none"> <li><b>HAL</b>: status</li> </ul>                                                                                                                                                                                                                                                                                             |
| <b>HAL_NAND_Write_Page_8b</b>     |                                                                                                                                                                                                                                                                                                                                                                  |
| Function name                     | <b>HAL_StatusTypeDef HAL_NAND_Write_Page_8b<br/>(NAND_HandleTypeDef * hndl, NAND_AddressTypeDef * pAddress, uint8_t * pBuffer, uint32_t NumPageToWrite)</b>                                                                                                                                                                                                      |
| Function description              | Write Page(s) to NAND memory block (8-bits addressing)                                                                                                                                                                                                                                                                                                           |
| Parameters                        | <ul style="list-style-type: none"> <li><b>hndl</b>: pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li><b>pAddress</b>: pointer to NAND address structure</li> <li><b>pBuffer</b>: pointer to source buffer to write</li> <li><b>NumPageToWrite</b>: : number of pages to write to block</li> </ul> |
| Return values                     | <ul style="list-style-type: none"> <li><b>HAL</b>: status</li> </ul>                                                                                                                                                                                                                                                                                             |
| <b>HAL_NAND_Read_SpareArea_8b</b> |                                                                                                                                                                                                                                                                                                                                                                  |
| Function name                     | <b>HAL_StatusTypeDef HAL_NAND_Read_SpareArea_8b<br/>(NAND_HandleTypeDef * hndl, NAND_AddressTypeDef * pAddress, uint8_t * pBuffer, uint32_t NumSpareAreaToRead)</b>                                                                                                                                                                                              |
| Function description              | Read Spare area(s) from NAND memory.                                                                                                                                                                                                                                                                                                                             |
| Parameters                        | <ul style="list-style-type: none"> <li><b>hndl</b>: pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li><b>pAddress</b>: pointer to NAND address structure</li> <li><b>pBuffer</b>: pointer to source buffer to write</li> </ul>                                                                     |

- **NumSpareAreaToRead:** Number of spare area to read
- Return values • **HAL:** status

### **HAL\_NAND\_Write\_SpareArea\_8b**

|                      |                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NAND_Write_SpareArea_8b<br/>(NAND_HandleTypeDef * hnand, NAND_AddressTypeDef *<br/>pAddress, uint8_t * pBuffer, uint32_t NumSpareAreaTowrite)</b>                                                                                                                                                                                                           |
| Function description | Write Spare area(s) to NAND memory.                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li>• <b>pAddress:</b> pointer to NAND address structure</li> <li>• <b>pBuffer:</b> pointer to source buffer to write</li> <li>• <b>NumSpareAreaTowrite:</b> : number of spare areas to write to block</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                 |

### **HAL\_NAND\_Read\_Page\_16b**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NAND_Read_Page_16b<br/>(NAND_HandleTypeDef * hnand, NAND_AddressTypeDef *<br/>pAddress, uint16_t * pBuffer, uint32_t NumPageToRead)</b>                                                                                                                                                                                                                                           |
| Function description | Read Page(s) from NAND memory block (16-bits addressing)                                                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li>• <b>pAddress:</b> pointer to NAND address structure</li> <li>• <b>pBuffer:</b> pointer to destination read buffer. pBuffer should be 16bits aligned</li> <li>• <b>NumPageToRead:</b> number of pages to read from block</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                       |

### **HAL\_NAND\_Write\_Page\_16b**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NAND_Write_Page_16b<br/>(NAND_HandleTypeDef * hnand, NAND_AddressTypeDef *<br/>pAddress, uint16_t * pBuffer, uint32_t NumPageToWrite)</b>                                                                                                                                                                                                                                          |
| Function description | Write Page(s) to NAND memory block (16-bits addressing)                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li>• <b>pAddress:</b> pointer to NAND address structure</li> <li>• <b>pBuffer:</b> pointer to source buffer to write. pBuffer should be 16bits aligned</li> <li>• <b>NumPageToWrite:</b> : number of pages to write to block</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                        |

**HAL\_NAND\_Read\_SpareArea\_16b**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NAND_Read_SpareArea_16b<br/>(NAND_HandleTypeDef * hñand, NAND_AddressTypeDef *<br/>pAddress, uint16_t * pBuffer, uint32_t NumSpareAreaToRead)</b>                                                                                                                                                                                                                                |
| Function description | Read Spare area(s) from NAND memory (16-bits addressing)                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hñand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li>• <b>pAddress:</b> pointer to NAND address structure</li> <li>• <b>pBuffer:</b> pointer to source buffer to write. pBuffer should be 16bits aligned.</li> <li>• <b>NumSpareAreaToRead:</b> Number of spare area to read</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                    |

**HAL\_NAND\_Write\_SpareArea\_16b**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NAND_Write_SpareArea_16b<br/>(NAND_HandleTypeDef * hñand, NAND_AddressTypeDef *<br/>pAddress, uint16_t * pBuffer, uint32_t NumSpareAreaTowrite)</b>                                                                                                                                                                                                                                            |
| Function description | Write Spare area(s) to NAND memory (16-bits addressing)                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hñand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li>• <b>pAddress:</b> pointer to NAND address structure</li> <li>• <b>pBuffer:</b> pointer to source buffer to write. pBuffer should be 16bits aligned.</li> <li>• <b>NumSpareAreaTowrite:</b> : number of spare areas to write to block</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                  |

**HAL\_NAND\_Erase\_Block**

|                      |                                                                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NAND_Erase_Block<br/>(NAND_HandleTypeDef * hñand, NAND_AddressTypeDef *<br/>pAddress)</b>                                                                                                                     |
| Function description | NAND memory Block erase.                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hñand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li>• <b>pAddress:</b> pointer to NAND address structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                 |

**HAL\_NAND\_Read\_Status**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_NAND_Read_Status (NAND_HandleTypeDef *<br/>hñand)</b>                                                                                                    |
| Function description | NAND memory read status.                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hñand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>NAND:</b> status</li> </ul>                                                                                                  |

**HAL\_NAND\_Address\_Inc**

|                      |                                                                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_NAND_Address_Inc (NAND_HandleTypeDef * hnand, NAND_HandleTypeDef * pAddress)</code>                                                                                                                                                                                                            |
| Function description | Increment the NAND memory address.                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li>• <b>pAddress:</b> pointer to NAND address structure</li> </ul>                                                                            |
| Return values        | <ul style="list-style-type: none"> <li>• <b>The:</b> new status of the increment address operation. It can be: <ul style="list-style-type: none"> <li>– NAND_VALID_ADDRESS: When the new address is valid address</li> <li>– NAND_INVALID_ADDRESS: When the new address is invalid address</li> </ul> </li> </ul> |

**HAL\_NAND\_ECC\_Enable**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_NAND_ECC_Enable (NAND_HandleTypeDef * hnand)</code>                                                                                          |
| Function description | Enables dynamically NAND ECC feature.                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                   |

**HAL\_NAND\_ECC\_Disable**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_NAND_ECC_Disable (NAND_HandleTypeDef * hnand)</code>                                                                                         |
| Function description | Disables dynamically FMC_NAND ECC feature.                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                   |

**HAL\_NAND\_GetECC**

|                      |                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_NAND_GetECC (NAND_HandleTypeDef * hnand, uint32_t * ECCval, uint32_t Timeout)</code>                                                                                                                                                           |
| Function description | Disables dynamically NAND ECC feature.                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnand:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li> <li>• <b>ECCval:</b> pointer to ECC value</li> <li>• <b>Timeout:</b> maximum timeout to wait</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                     |

**HAL\_NAND\_GetState**

|                      |                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_NAND_StateTypeDef HAL_NAND_GetState<br/>(NAND_HandleTypeDef * hndl)</b>                                                                                        |
| Function description | return the NAND state                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hndl:</b> pointer to a NAND_HandleTypeDef structure that contains the configuration information for NAND module.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> state</li></ul>                                                                                                   |

## 31.3 NAND Firmware driver defines

### 31.3.1 NAND

*NAND Exported Macros*

**\_HAL\_NAND\_RESET\_HANDLE\_STATE** **Description:**

- Reset NAND handle state.

**Parameters:**

- **\_HANDLE\_**: specifies the NAND handle.

**Return value:**

- None

## 32 HAL NOR Generic Driver

### 32.1 NOR Firmware driver registers structures

#### 32.1.1 NOR\_IDTypeDef

##### Data Fields

- `uint16_t Manufacturer_Code`
- `uint16_t Device_Code1`
- `uint16_t Device_Code2`
- `uint16_t Device_Code3`

##### Field Documentation

- `uint16_t NOR_IDTypeDef::Manufacturer_Code`  
Defines the device's manufacturer code used to identify the memory
- `uint16_t NOR_IDTypeDef::Device_Code1`
- `uint16_t NOR_IDTypeDef::Device_Code2`
- `uint16_t NOR_IDTypeDef::Device_Code3`  
Defines the device's codes used to identify the memory. These codes can be accessed by performing read operations with specific control signals and addresses set. They can also be accessed by issuing an Auto Select command.

#### 32.1.2 NOR\_CFITypeDef

##### Data Fields

- `uint16_t CFI_1`
- `uint16_t CFI_2`
- `uint16_t CFI_3`
- `uint16_t CFI_4`

##### Field Documentation

- `uint16_t NOR_CFITypeDef::CFI_1`
- `uint16_t NOR_CFITypeDef::CFI_2`
- `uint16_t NOR_CFITypeDef::CFI_3`
- `uint16_t NOR_CFITypeDef::CFI_4`  
Defines the information stored in the memory's Common flash interface which contains a description of various electrical and timing parameters, density information and functions supported by the memory.

#### 32.1.3 NOR\_HandleTypeDefDef

##### Data Fields

- `FMC_NORSRAM_TypeDef * Instance`
- `FMC_NORSRAM_EXTENDED_TypeDef * Extended`
- `FMC_NORSRAM_InitTypeDef Init`
- `HAL_LockTypeDef Lock`
- `__IO HAL_NOR_StateTypeDef State`

##### Field Documentation

- `FMC_NORSRAM_TypeDef* NOR_HandleTypeDefDef::Instance`  
Register base address

- ***FMC\_NORSRAM\_EXTENDED\_TypeDef\* NOR\_HandleTypeDef::Extended***  
Extended mode register base address
- ***FMC\_NORSRAM\_InitTypeDef NOR\_HandleTypeDef::Init***  
NOR device control configuration parameters
- ***HAL\_LockTypeDef NOR\_HandleTypeDef::Lock***  
NOR locking object
- ***\_\_IO HAL\_NOR\_StateTypeDef NOR\_HandleTypeDef::State***  
NOR device access state

## 32.2 NOR Firmware driver API description

### 32.2.1 How to use this driver

This driver is a generic layered driver which contains a set of APIs used to control NOR flash memories. It uses the FMC layer functions to interface with NOR devices. This driver is used as follows:

- NOR flash memory configuration sequence using the function HAL\_NOR\_Init() with control and timing parameters for both normal and extended mode.
- Read NOR flash memory manufacturer code and device IDs using the function HAL\_NOR\_Read\_ID(). The read information is stored in the NOR\_ID\_TypeDef structure declared by the function caller.
- Access NOR flash memory by read/write data unit operations using the functions HAL\_NOR\_Read(), HAL\_NOR\_Program().
- Perform NOR flash erase block/chip operations using the functions HAL\_NOR\_Erase\_Block() and HAL\_NOR\_Erase\_Chip().
- Read the NOR flash CFI (common flash interface) IDs using the function HAL\_NOR\_Read\_CFI(). The read information is stored in the NOR\_CFI\_TypeDef structure declared by the function caller.
- You can also control the NOR device by calling the control APIs HAL\_NOR\_WriteOperation\_Enable() / HAL\_NOR\_WriteOperation\_Disable() to respectively enable/disable the NOR write operation
- You can monitor the NOR device HAL state by calling the function HAL\_NOR\_GetState()



This driver is a set of generic APIs which handle standard NOR flash operations. If a NOR flash device contains different operations and/or implementations, it should be implemented separately.

### NOR HAL driver macros list

Below the list of most used macros in NOR HAL driver.

- NOR\_WRITE: NOR memory write data to specified address

### 32.2.2 NOR Initialization and de\_initialization functions

This section provides functions allowing to initialize/de-initialize the NOR memory

This section contains the following APIs:

- [\*\*HAL\\_NOR\\_Init\(\)\*\*](#)
- [\*\*HAL\\_NOR\\_DeInit\(\)\*\*](#)
- [\*\*HAL\\_NOR\\_MspInit\(\)\*\*](#)
- [\*\*HAL\\_NOR\\_MspDeInit\(\)\*\*](#)
- [\*\*HAL\\_NOR\\_MspWait\(\)\*\*](#)

### 32.2.3 NOR Input and Output functions

This section provides functions allowing to use and control the NOR memory

This section contains the following APIs:

- [\*HAL\\_NOR\\_Read\\_ID\(\)\*](#)
- [\*HAL\\_NOR\\_ReturnToReadMode\(\)\*](#)
- [\*HAL\\_NOR\\_Read\(\)\*](#)
- [\*HAL\\_NOR\\_Program\(\)\*](#)
- [\*HAL\\_NOR\\_ReadBuffer\(\)\*](#)
- [\*HAL\\_NOR\\_ProgramBuffer\(\)\*](#)
- [\*HAL\\_NOR\\_Erase\\_Block\(\)\*](#)
- [\*HAL\\_NOR\\_Erase\\_Chip\(\)\*](#)
- [\*HAL\\_NOR\\_Read\\_CFI\(\)\*](#)

### 32.2.4 NOR Control functions

This subsection provides a set of functions allowing to control dynamically the NOR interface.

This section contains the following APIs:

- [\*HAL\\_NOR\\_WriteOperation\\_Enable\(\)\*](#)
- [\*HAL\\_NOR\\_WriteOperation\\_Disable\(\)\*](#)

### 32.2.5 NOR State functions

This subsection permits to get in run-time the status of the NOR controller and the data flow.

This section contains the following APIs:

- [\*HAL\\_NOR\\_GetState\(\)\*](#)
- [\*HAL\\_NOR\\_GetStatus\(\)\*](#)

### 32.2.6 Detailed description of functions

#### **HAL\_NOR\_Init**

Function name      **HAL\_StatusTypeDef HAL\_NOR\_Init (NOR\_HandleTypeDef \*  
hnor, FMC\_NORSRAM\_TimingTypeDef \* Timing,  
FMC\_NORSRAM\_TimingTypeDef \* ExtTiming)**

Function description      Perform the NOR memory Initialization sequence.

Parameters     
 

- **hnor:** pointer to a NOR\_HandleTypeDef structure that contains the configuration information for NOR module.
- **Timing:** pointer to NOR control timing structure
- **ExtTiming:** pointer to NOR extended mode timing structure

Return values     
 

- **HAL:** status

#### **HAL\_NOR\_DeInit**

Function name      **HAL\_StatusTypeDef HAL\_NOR\_DeInit (NOR\_HandleTypeDef \*  
hnor)**

Function description      Perform NOR memory De-Initialization sequence.

---

|               |                                                                                                                                                                       |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                |

### HAL\_NOR\_MspInit

|                      |                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_NOR_MspInit (NOR_HandleTypeDef * hnor)</b>                                                                                                                |
| Function description | NOR MSP Init.                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                       |

### HAL\_NOR\_MspDeInit

|                      |                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_NOR_MspDeInit (NOR_HandleTypeDef * hnor)</b>                                                                                                              |
| Function description | NOR MSP DeInit.                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                       |

### HAL\_NOR\_MspWait

|                      |                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_NOR_MspWait (NOR_HandleTypeDef * hnor, uint32_t Timeout)</b>                                                                                                                                               |
| Function description | NOR MSP Wait fro Ready/Busy signal.                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> <li>• <b>Timeout:</b> Maximum timeout value</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                        |

### HAL\_NOR\_Read\_ID

|                      |                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_Read_ID (NOR_HandleTypeDef * hnor, NOR_IDTypeDef * pNOR_ID)</b>                                                                                                                                 |
| Function description | Read NOR flash IDs.                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> <li>• <b>pNOR_ID:</b> pointer to NOR ID structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                       |

### HAL\_NOR\_ReturnToReadMode

|                      |                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_ReturnToReadMode (NOR_HandleTypeDef * hnor)</b>                                   |
| Function description | Returns the NOR memory to Read mode.                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that</li> </ul> |

contains the configuration information for NOR module.

|               |                                                                      |
|---------------|----------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul> |
|---------------|----------------------------------------------------------------------|

### HAL\_NOR\_Read

|                      |                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_Read (NOR_HandleTypeDef * hnor, uint32_t * pAddress, uint16_t * pData)</b>                                                                                                                                                             |
| Function description | Read data from NOR memory.                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> <li><b>pAddress:</b> pointer to Device address</li> <li><b>pData:</b> pointer to read data</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                |

### HAL\_NOR\_Program

|                      |                                                                                                                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_Program (NOR_HandleTypeDef * hnor, uint32_t * pAddress, uint16_t * pData)</b>                                                                                                                                                       |
| Function description | Program data to NOR memory.                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> <li><b>pAddress:</b> Device address</li> <li><b>pData:</b> pointer to the data to write</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                             |

### HAL\_NOR\_ReadBuffer

|                      |                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_ReadBuffer (NOR_HandleTypeDef * hnor, uint32_t uwAddress, uint16_t * pData, uint32_t uwBufferSize)</b>                                                                                                                                                                                                                                                               |
| Function description | Reads a block of data from the FMC NOR memory.                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li><b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> <li><b>uwAddress:</b> NOR memory internal address to read from.</li> <li><b>pData:</b> pointer to the buffer that receives the data read from the NOR memory.</li> <li><b>uwBufferSize:</b> number of Half word to read.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                              |

### HAL\_NOR\_ProgramBuffer

|                      |                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_ProgramBuffer (NOR_HandleTypeDef * hnor, uint32_t uwAddress, uint16_t * pData, uint32_t uwBufferSize)</b>                              |
| Function description | Writes a half-word buffer to the FMC NOR memory.                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li><b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> </ul> |

|               |                                                                                                                                                                                                                                                               |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>• <b>uwAddress:</b> NOR memory internal address from which the data</li> <li>• <b>pData:</b> pointer to source data buffer.</li> <li>• <b>uwBufferSize:</b> number of Half words to write.</li> </ul>                  |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                        |
| Notes         | <ul style="list-style-type: none"> <li>• Some NOR memory need Address aligned to xx bytes (can be aligned to 64 bytes boundary for example).</li> <li>• The maximum buffer size allowed is NOR memory dependent (can be 64 Bytes max for example).</li> </ul> |

### HAL\_NOR\_Erase\_Block

|                      |                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_Erase_Block<br/>(NOR_HandleTypeDef * hnor, uint32_t BlockAddress, uint32_t Address)</b>                                                                                                                                                   |
| Function description | Erase the specified block of the NOR memory.                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> <li>• <b>BlockAddress:</b> Block to erase address</li> <li>• <b>Address:</b> Device address</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                 |

### HAL\_NOR\_Erase\_Chip

|                      |                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_Erase_Chip<br/>(NOR_HandleTypeDef * hnor, uint32_t Address)</b>                                                                                                                    |
| Function description | Erase the entire NOR chip.                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> <li>• <b>Address:</b> Device address</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                          |

### HAL\_NOR\_Read\_CFI

|                      |                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_Read_CFI<br/>(NOR_HandleTypeDef * hnor, NOR_CFITypeDef * pNOR_CFI)</b>                                                                                                                                |
| Function description | Read NOR flash CFI IDs.                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li> <li>• <b>pNOR_CFI:</b> pointer to NOR CFI IDs structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                             |

### HAL\_NOR\_WriteOperation\_Enable

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_WriteOperation_Enable<br/>(NOR_HandleTypeDef * hnor)</b> |
| Function description | Enables dynamically NOR write operation.                                              |

|               |                                                                                                                                                                     |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"><li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li></ul> |
| Return values | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                |

### **HAL\_NOR\_WriteOperation\_Disable**

|                      |                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_NOR_WriteOperation_Disable<br/>(NOR_HandleTypeDef * hnor)</b>                                                                              |
| Function description | Disables dynamically NOR write operation.                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                |

### **HAL\_NOR\_GetState**

|                      |                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_NOR_StateTypeDef HAL_NOR_GetState<br/>(NOR_HandleTypeDef * hnor)</b>                                                                                         |
| Function description | return the NOR controller state                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>NOR:</b> controller state</li></ul>                                                                                      |

### **HAL\_NOR\_GetStatus**

|                      |                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_NOR_StatusTypeDef HAL_NOR_GetStatus<br/>(NOR_HandleTypeDef * hnor, uint32_t Address, uint32_t Timeout)</b>                                                                                                                                              |
| Function description | Returns the NOR operation status.                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hnor:</b> pointer to a NOR_HandleTypeDef structure that contains the configuration information for NOR module.</li><li>• <b>Address:</b> Device address</li><li>• <b>Timeout:</b> NOR programming Timeout</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>NOR_Status:</b> The returned value can be:<br/>HAL_NOR_STATUS_SUCCESS,<br/>HAL_NOR_STATUS_ERROR or<br/>HAL_NOR_STATUS_TIMEOUT</li></ul>                                                                             |

## 32.3 NOR Firmware driver defines

### 32.3.1 NOR

#### *NOR Exported Macros*

`_HAL_NOR_RESET_HANDLE_STATE` **Description:**

- Reset NOR handle state.

**Parameters:**

- `_HANDLE_`: NOR handle

**Return value:**

- None

## 33 HAL OPAMP Generic Driver

### 33.1 OPAMP Firmware driver registers structures

#### 33.1.1 OPAMP\_InitTypeDef

##### Data Fields

- *uint32\_t Mode*
- *uint32\_t InvertingInput*
- *uint32\_t NonInvertingInput*
- *uint32\_t TimerControlledMuxmode*
- *uint32\_t InvertingInputSecondary*
- *uint32\_t NonInvertingInputSecondary*
- *uint32\_t PgaConnect*
- *uint32\_t PgaGain*
- *uint32\_t UserTrimming*
- *uint32\_t TrimmingValueP*
- *uint32\_t TrimmingValueN*

##### Field Documentation

- ***uint32\_t OPAMP\_InitTypeDef::Mode***  
Specifies the OPAMP mode This parameter must be a value of [OPAMP\\_Mode](#) mode is either Standalone, - Follower or PGA
- ***uint32\_t OPAMP\_InitTypeDef::InvertingInput***  
Specifies the inverting input in Standalone & Pga modesIn Standalone mode: i.e when mode is OPAMP\_STANDALONE\_MODE This parameter must be a value of [OPAMP\\_InvertingInput](#) InvertingInput is either VM0 or VM1In PGA mode: i.e when mode is OPAMP\_PGA\_MODE & in Follower mode i.e when mode is OPAMP\_FOLLOWER\_MODE This parameter is Not Applicable
- ***uint32\_t OPAMP\_InitTypeDef::NonInvertingInput***  
Specifies the non inverting input of the opamp: This parameter must be a value of [OPAMP\\_NonInvertingInput](#) NonInvertingInput is either VP0, VP1, VP2 or VP3
- ***uint32\_t OPAMP\_InitTypeDef::TimerControlledMuxmode***  
Specifies if the Timer controlled Mux mode is enabled or disabled This parameter must be a value of [OPAMP\\_TimerControlledMuxmode](#)
- ***uint32\_t OPAMP\_InitTypeDef::InvertingInputSecondary***  
Specifies the inverting input (secondary) of the opamp when TimerControlledMuxmode is enabled i.e. when TimerControlledMuxmode is OPAMP\_TIMERCONTROLLEDMUXMODE\_ENABLEIn Standalone mode: i.e when mode is OPAMP\_STANDALONE\_MODE This parameter must be a value of [OPAMP\\_InvertingInputSecondary](#) InvertingInputSecondary is either VM0 or VM1In PGA mode: i.e when mode is OPAMP\_PGA\_MODE & in Follower mode i.e when mode is OPAMP\_FOLLOWER\_MODE This parameter is Not Applicable
- ***uint32\_t OPAMP\_InitTypeDef::NonInvertingInputSecondary***  
Specifies the non inverting input (secondary) of the opamp when TimerControlledMuxmode is enabled i.e. when TimerControlledMuxmode is OPAMP\_TIMERCONTROLLEDMUXMODE\_ENABLE This parameter must be a value of [OPAMP\\_NonInvertingInputSecondary](#) NonInvertingInput is either VP0, VP1, VP2 or VP3
- ***uint32\_t OPAMP\_InitTypeDef::PgaConnect***  
Specifies the inverting pin in PGA mode i.e. when mode is OPAMP\_PGA\_MODE This

- parameter must be a value of ***OPAMP\_PgaConnect*** Either: not connected, connected to VM0, connected to VM1 (VM0 or VM1 are typically used for external filtering)
- ***uint32\_t OPAMP\_InitTypeDef::PgaGain***  
Specifies the gain in PGA mode i.e. when mode is OPAMP\_PGA\_MODE. This parameter must be a value of ***OPAMP\_PgaGain*** (2U, 4U, 8 or 16U )
  - ***uint32\_t OPAMP\_InitTypeDef::UserTrimming***  
Specifies the trimming mode This parameter must be a value of ***OPAMP\_UserTrimming*** UserTrimming is either factory or user trimming
  - ***uint32\_t OPAMP\_InitTypeDef::TrimmingValueP***  
Specifies the offset trimming value (PMOS) i.e. when UserTrimming is OPAMP\_TRIMMING\_USER. This parameter must be a number between Min\_Data = 1 and Max\_Data = 31U
  - ***uint32\_t OPAMP\_InitTypeDef::TrimmingValueN***  
Specifies the offset trimming value (NMOS) i.e. when UserTrimming is OPAMP\_TRIMMING\_USER. This parameter must be a number between Min\_Data = 1 and Max\_Data = 31U

### 33.1.2 OPAMP\_HandleTypeDef

#### Data Fields

- ***OPAMP\_TypeDef \* Instance***
- ***OPAMP\_InitTypeDef Init***
- ***HAL\_StatusTypeDef Status***
- ***HAL\_LockTypeDef Lock***
- ***\_\_IO HAL\_OPAMP\_StateTypeDef State***

#### Field Documentation

- ***OPAMP\_TypeDef\* OPAMP\_HandleTypeDef::Instance***  
OPAMP instance's registers base address
- ***OPAMP\_InitTypeDef OPAMP\_HandleTypeDef::Init***  
OPAMP required parameters
- ***HAL\_StatusTypeDef OPAMP\_HandleTypeDef::Status***  
OPAMP peripheral status
- ***HAL\_LockTypeDef OPAMP\_HandleTypeDef::Lock***  
Locking object
- ***\_\_IO HAL\_OPAMP\_StateTypeDef OPAMP\_HandleTypeDef::State***  
OPAMP communication state

## 33.2 OPAMP Firmware driver API description

### 33.2.1 OPAMP Peripheral Features

The device integrates up to 4 operational amplifiers OPAMP1, OPAMP2, OPAMP3 and OPAMP4:

1. The OPAMP(s) provides several exclusive running modes.
  - Standalone mode
  - Programmable Gain Amplifier (PGA) mode (Resistor feedback output)
  - Follower mode
2. The OPAMP(s) provide(s) calibration capabilities.
  - Calibration aims at correcting some offset for running mode.
  - The OPAMP uses either factory calibration settings OR user defined calibration (trimming) settings (i.e. trimming mode).
  - The user defined settings can be figured out using self calibration handled by HAL\_OPAMP\_SelfCalibrate, HAL\_OPAMPEx\_SelfCalibrateAll

- HAL\_OPAMP\_SelfCalibrate:
  - Runs automatically the calibration in 2 steps. (90U% of VDDA for NMOS transistors, 10U% of VDDA for PMOS transistors). (As OPAMP is Rail-to-rail input/output, these 2 steps calibration is appropriate and enough in most cases).
  - Enables the user trimming mode
  - Updates the init structure with trimming values with fresh calibration results. The user may store the calibration results for larger (ex monitoring the trimming as a function of temperature for instance)
  - for STM32F3 devices having 2 or 4 OPAMPs HAL\_OPAMPEx\_SelfCalibrateAll runs calibration of 2 or 4 OPAMPs in parallel.
3. For any running mode, an additional Timer-controlled Mux (multiplexer) mode can be set on top.
    - Timer-controlled Mux mode allows Automatic switching between inverting and non-inverting input.
    - Hence on top of defaults (primary) inverting and non-inverting inputs, the user shall select secondary inverting and non inverting inputs.
    - TIM1 CC6 provides the alternate switching tempo between defaults (primary) and secondary inputs.
  4. Running mode: Standalone mode
    - Gain is set externally (gain depends on external loads).
    - Follower mode also possible externally by connecting the inverting input to the output.
  5. Running mode: Follower mode
    - No Inverting Input is connected.
  6. Running mode: Programmable Gain Amplifier (PGA) mode (Resistor feedback output)
    - The OPAMP(s) output(s) can be internally connected to resistor feedback output.
    - OPAMP gain is either 2U, 4U, 8 or 16.

### 33.2.2 How to use this driver

#### Calibration

To run the opamp calibration self calibration:

1. Start calibration using HAL\_OPAMP\_SelfCalibrate. Store the calibration results.

#### Running mode

To use the opamp, perform the following steps:

1. Fill in the HAL\_OPAMP\_MspInit() to
  - Configure the opamp input AND output in analog mode using HAL\_GPIO\_Init() to map the opamp output to the GPIO pin.
2. Configure the opamp using HAL\_OPAMP\_Init() function:
  - Select the mode
  - Select the inverting input
  - Select the non-inverting input
  - Select if the Timer controlled Mux mode is enabled/disabled
  - If the Timer controlled Mux mode is enabled, select the secondary inverting input
  - If the Timer controlled Mux mode is enabled, Select the secondary non-inverting input
  - If PGA mode is enabled, Select if inverting input is connected.
  - Select either factory or user defined trimming mode.

- If the user defined trimming mode is enabled, select PMOS & NMOS trimming values (typ. settings returned by HAL\_OPAMP\_SelfCalibrate function).
- 3. Enable the opamp using HAL\_OPAMP\_Start() function.
- 4. Disable the opamp using HAL\_OPAMP\_Stop() function.
- 5. Lock the opamp in running mode using HAL\_OPAMP\_Lock() function. From then The configuration can be modified
  - After HW reset
  - OR thanks to HAL\_OPAMP\_MspDelnit called (user defined) from HAL\_OPAMP\_Delinit.

### **Running mode: change of configuration while OPAMP ON**

To Re-configure OPAMP when OPAMP is ON (change on the fly)

1. If needed, Fill in the HAL\_OPAMP\_MsplInit()
  - This is the case for instance if you wish to use new OPAMP I/O
2. Configure the opamp using HAL\_OPAMP\_Init() function:
  - As in configure case, selects first the parameters you wish to modify.

### **33.2.3 Initialization and de-initialization functions**

This section provides functions allowing to:

This section contains the following APIs:

- [\*HAL\\_OPAMP\\_Init\(\)\*](#)
- [\*HAL\\_OPAMP\\_Delinit\(\)\*](#)
- [\*HAL\\_OPAMP\\_MsplInit\(\)\*](#)
- [\*HAL\\_OPAMP\\_MspDelinit\(\)\*](#)

### **33.2.4 IO operation functions**

This subsection provides a set of functions allowing to manage the OPAMP data transfers.

This section contains the following APIs:

- [\*HAL\\_OPAMP\\_Start\(\)\*](#)
- [\*HAL\\_OPAMP\\_Stop\(\)\*](#)
- [\*HAL\\_OPAMP\\_SelfCalibrate\(\)\*](#)

### **33.2.5 Peripheral Control functions**

This subsection provides a set of functions allowing to control the OPAMP data transfers.

This section contains the following APIs:

- [\*HAL\\_OPAMP\\_Lock\(\)\*](#)

### **33.2.6 Peripheral State functions**

This subsection permit to get in run-time the status of the peripheral and the data flow.

This section contains the following APIs:

- [\*HAL\\_OPAMP\\_GetState\(\)\*](#)
- [\*HAL\\_OPAMP\\_GetTrimOffset\(\)\*](#)

### 33.2.7 Detailed description of functions

#### HAL\_OPAMP\_Init

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_OPAMP_Init<br/>(OPAMP_HandleTypeDef * hopamp)</b>                                                                                                   |
| Function description | Initializes the OPAMP according to the specified parameters in the OPAMP_InitTypeDef and create the associated handle.                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hopamp:</b> OPAMP handle</li> </ul>                                                                                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• If the selected opamp is locked, initialization can't be performed. To unlock the configuration, perform a system reset.</li> </ul> |

#### HAL\_OPAMP\_DeInit

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_OPAMP_DeInit<br/>(OPAMP_HandleTypeDef * hopamp)</b>                                                                                                       |
| Function description | Deinitializes the OPAMP peripheral.                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hopamp:</b> OPAMP handle</li> </ul>                                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• Deinitialization can't be performed if the OPAMP configuration is locked. To unlock the configuration, perform a system reset.</li> </ul> |

#### HAL\_OPAMP\_MspInit

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <b>void HAL_OPAMP_MspInit (OPAMP_HandleTypeDef * hopamp)</b>                    |
| Function description | Initializes the OPAMP MSP.                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hopamp:</b> OPAMP handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                 |

#### HAL\_OPAMP\_MspDeInit

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <b>void HAL_OPAMP_MspDeInit (OPAMP_HandleTypeDef * hopamp)</b>                  |
| Function description | Deinitializes OPAMP MSP.                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hopamp:</b> OPAMP handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                 |

#### HAL\_OPAMP\_Start

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_OPAMP_Start<br/>(OPAMP_HandleTypeDef * hopamp)</b> |
| Function description | Start the opamp.                                                            |

---

|               |                                                                             |
|---------------|-----------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"><li><b>hopamp:</b> OPAMP handle</li></ul> |
| Return values | <ul style="list-style-type: none"><li><b>HAL:</b> status</li></ul>          |

### HAL\_OPAMP\_Stop

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_OPAMP_Stop<br/>(OPAMP_HandleTypeDef * hopamp)</b>  |
| Function description | Stop the opamp.                                                             |
| Parameters           | <ul style="list-style-type: none"><li><b>hopamp:</b> OPAMP handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li><b>HAL:</b> status</li></ul>          |

### HAL\_OPAMP\_SelfCalibrate

|                      |                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_OPAMP_SelfCalibrate<br/>(OPAMP_HandleTypeDef * hopamp)</b>                                                                           |
| Function description | Run the self calibration of one OPAMP.                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"><li><b>hopamp:</b> handle</li></ul>                                                                                         |
| Return values        | <ul style="list-style-type: none"><li><b>Updated:</b> offset trimming values (PMOS &amp; NMOS), user trimming is enabled</li><li><b>HAL:</b> status</li></ul> |
| Notes                | <ul style="list-style-type: none"><li>Calibration runs about 25 ms.</li></ul>                                                                                 |

### HAL\_OPAMP\_Lock

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_OPAMP_Lock<br/>(OPAMP_HandleTypeDef * hopamp)</b>  |
| Function description | Lock the selected opamp configuration.                                      |
| Parameters           | <ul style="list-style-type: none"><li><b>hopamp:</b> OPAMP handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li><b>HAL:</b> status</li></ul>          |

### HAL\_OPAMP\_GetState

|                      |                                                                                     |
|----------------------|-------------------------------------------------------------------------------------|
| Function name        | <b>HAL_OPAMP_StateTypeDef HAL_OPAMP_GetState<br/>(OPAMP_HandleTypeDef * hopamp)</b> |
| Function description | Return the OPAMP state.                                                             |
| Parameters           | <ul style="list-style-type: none"><li><b>hopamp:</b> OPAMP handle</li></ul>         |
| Return values        | <ul style="list-style-type: none"><li><b>HAL:</b> state</li></ul>                   |

### HAL\_OPAMP\_GetTrimOffset

|                      |                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>OPAMP_TrimmingValueTypeDef HAL_OPAMP_GetTrimOffset<br/>(OPAMP_HandleTypeDef * hopamp, uint32_t trimmingoffset)</b>               |
| Function description | Return the OPAMP factory trimming value.                                                                                            |
| Parameters           | <ul style="list-style-type: none"><li><b>hopamp:</b> OPAMP handle</li><li><b>trimmingoffset:</b> Trimming offset (P or N)</li></ul> |

- Return values**
- **Trimming:** value (P or N): range: 0->31 or OPAMP\_FACTORYTRIMMING\_DUMMY if trimming value is not available

### 33.3 OPAMP Firmware driver defines

#### 33.3.1 OPAMP

##### **OPAMP CSR init register Mask**

OPAMP\_CSR\_UPDATE\_PARAMETERS\_INIT\_MASK

##### **OPAMP Exported Macros**

`_HAL_OPAMP_RESET_HANDLE_STATE` **Description:**

- Reset OPAMP handle state.

##### **Parameters:**

- `_HANDLE_`: OPAMP handle.

##### **Return value:**

- None

##### **OPAMP Factory Trimming**

OPAMP\_FACTORYTRIMMING\_DUMMY Dummy trimming value

OPAMP\_FACTORYTRIMMING\_N Offset trimming N

OPAMP\_FACTORYTRIMMING\_P Offset trimming P

IS\_OPAMP\_FACTORYTRIMMING

##### **OPAMP Input**

OPAMP\_INPUT\_INVERTING Inverting input

OPAMP\_INPUT\_NONINVERTING Non inverting input

IS\_OPAMP\_INPUT

##### **OPAMP Inverting Input**

OPAMP\_INVERTINGINPUT\_IO0 inverting input connected to VM0

OPAMP\_INVERTINGINPUT\_IO1 inverting input connected to VM1

IS\_OPAMP\_INVERTING\_INPUT

##### **OPAMP Inverting Input Secondary**

OPAMP\_SEC\_INVERTINGINPUT\_IO0 VM0 (PC5 for OPAMP1 and OPAMP2, PB10 for OPAMP3 and OPAMP4) connected to OPAMPx inverting input

OPAMP\_SEC\_INVERTINGINPUT\_IO1 VM1 (PA3 for OPAMP1, PA5 for OPAMP2, PB2 for OPAMP3, PD8 for OPAMP4) connected to OPAMPx inverting input

IS\_OPAMP\_SEC\_INVERTINGINPUT

##### **OPAMP Mode**

OPAMP\_STANDALONE\_MODE standalone mode

|                                                   |                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| OPAMP_PGA_MODE                                    | PGA mode                                                                                                                    |
| OPAMP_FOLLOWER_MODE                               | follower mode                                                                                                               |
| <b>IS_OPAMP_FUNCTIONAL_NORMALMODE</b>             |                                                                                                                             |
| <b><i>OPAMP Non Inverting Input</i></b>           |                                                                                                                             |
| OPAMP_NONINVERTINGINPUT_IO0                       | VP0 (PA1 for OPAMP1, VP0 PA7 for OPAMP2, VP0 PB0 for OPAMP3, VP0 PB13 for OPAMP4) connected to OPAMPx non inverting input   |
| OPAMP_NONINVERTINGINPUT_IO1                       | VP1 (PA7 for OPAMP1, VP3 PD14 for OPAMP2, VP1 PB13 for OPAMP3, VP1 PD11 for OPAMP4) connected to OPAMPx non inverting input |
| OPAMP_NONINVERTINGINPUT_IO2                       | VP2 (PA3 for OPAMP1, VP2 PB0 for OPAMP2, VP2 PA1 for OPAMP3, VP3 PA4 for OPAMP4) connected to OPAMPx non inverting input    |
| OPAMP_NONINVERTINGINPUT_IO3                       | VP3 (PA5 for OPAMP1, VP1 PB14 for OPAMP2, VP3 PA5 for OPAMP3, VP2 PB11 for OPAMP4) connected to OPAMPx non inverting input  |
| <b>IS_OPAMP_NONINVERTING_INPUT</b>                |                                                                                                                             |
| <b><i>OPAMP Non Inverting Input Secondary</i></b> |                                                                                                                             |
| OPAMP_SEC_NONINVERTINGINPUT_IO0                   | VP0 (PA1 for OPAMP1, PA7 for OPAMP2, PB0 for OPAMP3, PB13 for OPAMP4) connected to OPAMPx non inverting input               |
| OPAMP_SEC_NONINVERTINGINPUT_IO1                   | VP1 (PA7 for OPAMP1, PD14 for OPAMP2, PB13 for OPAMP3, PD11 for OPAMP4) connected to OPAMPx non inverting input             |
| OPAMP_SEC_NONINVERTINGINPUT_IO2                   | VP2 (PA3 for OPAMP1, PB0 for OPAMP2, PA1 for OPAMP3, PA4 for OPAMP4) connected to OPAMPx non inverting input                |
| OPAMP_SEC_NONINVERTINGINPUT_IO3                   | VP3 (PA5 for OPAMP1, PB14 for OPAMP2, PA5 for OPAMP3, PB11 for OPAMP4) connected to OPAMPx non inverting input              |
| <b>IS_OPAMP_SEC_NONINVERTINGINPUT</b>             |                                                                                                                             |
| <b><i>OPAMP Pga Connect</i></b>                   |                                                                                                                             |
| OPAMP_PGA_CONNECT_INVERTINGINPUT_NO               | In PGA mode, the non inverting input is not connected                                                                       |
| OPAMP_PGA_CONNECT_INVERTINGINPUT_IO0              | In PGA mode, the non inverting input is connected to VM0                                                                    |
| OPAMP_PGA_CONNECT_INVERTINGINPUT_IO1              | In PGA mode, the non inverting input is connected to VM1                                                                    |
| <b>IS_OPAMP_PGACONNECT</b>                        |                                                                                                                             |
| <b><i>OPAMP Pga Gain</i></b>                      |                                                                                                                             |
| OPAMP_PGA_GAIN_2                                  | PGA gain = 2                                                                                                                |
| OPAMP_PGA_GAIN_4                                  | PGA gain = 4                                                                                                                |
| OPAMP_PGA_GAIN_8                                  | PGA gain = 8                                                                                                                |

OPAMP\_PGA\_GAIN\_16 PGA gain = 16

IS\_OPAMP\_PGA\_GAIN

***OPAMP Timer Controlled Mux mode***

OPAMP\_TIMERCONTROLLEDMUXMODE\_DISABLE Timer controlled Mux mode disabled

OPAMP\_TIMERCONTROLLEDMUXMODE\_ENABLE Timer controlled Mux mode enabled

IS\_OPAMP\_TIMERCONTROLLED\_MUXMODE

***OPAMP Trimming Value***

IS\_OPAMP\_TRIMMINGVALUE

***OPAMP User Trimming***

OPAMP\_TRIMMING\_FACTORY Factory trimming

OPAMP\_TRIMMING\_USER User trimming

IS\_OPAMP\_TRIMMING

OPAMP\_VREF\_NOTCONNECTEDTO\_ADC VREF not connected to ADC

OPAMP\_VREF\_CONNECTEDTO\_ADC VREF not connected to ADC

IS\_OPAMP\_ALLOPAMPVREF\_CONNECT

***OPAMP VREF***

OPAMP\_VREF\_3VDDA OPMAP Vref = 3.3U% VDDA

OPAMP\_VREF\_10VDDA OPMAP Vref = 10U% VDDA

OPAMP\_VREF\_50VDDA OPMAP Vref = 50U% VDDA

OPAMP\_VREF\_90VDDA OPMAP Vref = 90U% VDDA

IS\_OPAMP\_VREF

## 34 HAL OPAMP Extension Driver

### 34.1 OPAMPEx Firmware driver API description

#### 34.1.1 Detailed description of functions

##### **HAL\_OPAMPEx\_SelfCalibrateAll**

|                      |                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_OPAMPEx_SelfCalibrateAll<br/>(OPAMP_HandleTypeDef * hopamp1, OPAMP_HandleTypeDef * hopamp2,<br/>* hopamp3, OPAMP_HandleTypeDef * hopamp4,</b>      |
| Function description | Run the self calibration of 4 OPAMPs in parallel.                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hopamp1:</b> handle</li><li>• <b>hopamp2:</b> handle</li><li>• <b>hopamp3:</b> handle</li><li>• <b>hopamp4:</b> handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                        |
| Notes                | <ul style="list-style-type: none"><li>• Updated offset trimming values (PMOS &amp; NMOS), user trimming is enabled</li><li>• Calibration runs about 25 ms.</li></ul>        |

## 35 HAL PCCARD Generic Driver

### 35.1 PCCARD Firmware driver registers structures

#### 35.1.1 PCCARD\_HandleTypeDef

##### Data Fields

- *FMC\_PCCARD\_TypeDef \* Instance*
- *FMC\_PCCARD\_InitTypeDef Init*
- *\_\_IO HAL\_PCCARD\_StateTypeDef State*
- *HAL\_LockTypeDef Lock*

##### Field Documentation

- ***FMC\_PCCARD\_TypeDef\* PCCARD\_HandleTypeDef::Instance***  
Register base address for PCCARD device
- ***FMC\_PCCARD\_InitTypeDef PCCARD\_HandleTypeDef::Init***  
PCCARD device control configuration parameters
- ***\_\_IO HAL\_PCCARD\_StateTypeDef PCCARD\_HandleTypeDef::State***  
PCCARD device access state
- ***HAL\_LockTypeDef PCCARD\_HandleTypeDef::Lock***  
PCCARD Lock

### 35.2 PCCARD Firmware driver API description

#### 35.2.1 How to use this driver

This driver is a generic layered driver which contains a set of APIs used to control PCCARD/compact flash memories. It uses the FMC layer functions to interface with PCCARD devices. This driver is used for:

- PCCARD/compact flash memory configuration sequence using the function `HAL_PCCARD_Init()` with control and timing parameters for both common and attribute spaces.
- Read PCCARD/compact flash memory maker and device IDs using the function `HAL_PCCARD_Read_ID()`. The read information is stored in the `CompactFlash_ID` structure declared by the function caller.
- Access PCCARD/compact flash memory by read/write operations using the functions `HAL_PCCARD_Read_Sector()`/`HAL_PCCARD_Write_Sector()`, to read/write sector.
- Perform PCCARD/compact flash Reset chip operation using the function `HAL_PCCARD_Reset()`.
- Perform PCCARD/compact flash erase sector operation using the function `HAL_PCCARD_Erase_Sector()`.
- Read the PCCARD/compact flash status operation using the function `HAL_PCCARD_ReadStatus()`.
- You can monitor the PCCARD/compact flash device HAL state by calling the function `HAL_PCCARD_GetState()`



This driver is a set of generic APIs which handle standard PCCARD/compact flash operations. If a PCCARD/compact flash device contains different operations and/or implementations, it should be implemented separately.

### 35.2.2 PCCARD Initialization and de-initialization functions

This section provides functions allowing to initialize/de-initialize the PCCARD memory

This section contains the following APIs:

- [\*HAL\\_PCCARD\\_Init\(\)\*](#)
- [\*HAL\\_PCCARD\\_DelInit\(\)\*](#)
- [\*HAL\\_PCCARD\\_MspInit\(\)\*](#)
- [\*HAL\\_PCCARD\\_MspDelInit\(\)\*](#)

### 35.2.3 PCCARD Input Output and memory functions

This section provides functions allowing to use and control the PCCARD memory

This section contains the following APIs:

- [\*HAL\\_PCCARD\\_Read\\_ID\(\)\*](#)
- [\*HAL\\_PCCARD\\_Read\\_Sector\(\)\*](#)
- [\*HAL\\_PCCARD\\_Write\\_Sector\(\)\*](#)
- [\*HAL\\_PCCARD\\_Erase\\_Sector\(\)\*](#)
- [\*HAL\\_PCCARD\\_Reset\(\)\*](#)
- [\*HAL\\_PCCARD\\_IRQHandler\(\)\*](#)
- [\*HAL\\_PCCARD\\_ITCallback\(\)\*](#)

### 35.2.4 PCCARD Peripheral State functions

This subsection permits to get in run-time the status of the PCCARD controller and the data flow.

This section contains the following APIs:

- [\*HAL\\_PCCARD\\_GetState\(\)\*](#)
- [\*HAL\\_PCCARD\\_GetStatus\(\)\*](#)
- [\*HAL\\_PCCARD\\_ReadStatus\(\)\*](#)

### 35.2.5 Detailed description of functions

#### **HAL\_PCCARD\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_PCCARD_Init<br/>(PCCARD_HandleTypeDef * hpccard,<br/>FMC_NAND_PCC_TimingTypeDef * ComSpaceTiming,<br/>FMC_NAND_PCC_TimingTypeDef * AttSpaceTiming,<br/>FMC_NAND_PCC_TimingTypeDef * IOSpaceTiming)</code>                                                                                                                                        |
| Function description | Perform the PCCARD memory Initialization sequence.                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpccard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li> <li>• <b>ComSpaceTiming:</b> Common space timing structure</li> <li>• <b>AttSpaceTiming:</b> Attribute space timing structure</li> <li>• <b>IOSpaceTiming:</b> IO space timing structure</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                         |

**HAL\_PCCARD\_DelInit**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCCARD_DelInit<br/>(PCCARD_HandleTypeDef * hpccard)</b>                                                                                               |
| Function description | Perform the PCCARD memory De-initialization sequence.                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpccard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                         |

**HAL\_PCCARD\_MspInit**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCCARD_MspInit (PCCARD_HandleTypeDef * hpccard)</b>                                                                                                                |
| Function description | PCCARD MSP Init.                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpccard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                |

**HAL\_PCCARD\_MspDelInit**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCCARD_MspDelInit (PCCARD_HandleTypeDef * hpccard)</b>                                                                                                             |
| Function description | PCCARD MSP DelInit.                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpccard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                |

**HAL\_PCCARD\_Read\_ID**

|                      |                                                                                                                                                                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCCARD_Read_ID<br/>(PCCARD_HandleTypeDef * hpccard, uint8_t<br/>CompactFlash_ID, uint8_t * pStatus)</b>                                                                                                                                                                         |
| Function description | Read Compact Flash's ID.                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpccard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li> <li>• <b>CompactFlash_ID:</b> Compact flash ID structure.</li> <li>• <b>pStatus:</b> pointer to compact flash status</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                   |

**HAL\_PCCARD\_Write\_Sector**

|                      |                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCCARD_Write_Sector<br/>(PCCARD_HandleTypeDef * hpccard, uint16_t * pBuffer,<br/>uint16_t SectorAddress, uint8_t * pStatus)</b>                                                                                                                                                                                          |
| Function description | Write sector to PCCARD memory.                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpccard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li> <li>• <b>pBuffer:</b> pointer to source write buffer</li> <li>• <b>SectorAddress:</b> Sector address to write</li> <li>• <b>pStatus:</b> pointer to CF status</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                            |

**HAL\_PCCARD\_Read\_Sector**

|                      |                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCCARD_Read_Sector<br/>(PCCARD_HandleTypeDef * hpccard, uint16_t * pBuffer,<br/>uint16_t SectorAddress, uint8_t * pStatus)</b>                                                                                                                                                                                              |
| Function description | Read sector from PCCARD memory.                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpccard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li> <li>• <b>pBuffer:</b> pointer to destination read buffer</li> <li>• <b>SectorAddress:</b> Sector address to read</li> <li>• <b>pStatus:</b> pointer to CF status</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                               |

**HAL\_PCCARD\_Erase\_Sector**

|                      |                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCCARD_Erase_Sector<br/>(PCCARD_HandleTypeDef * hpccard, uint16_t SectorAddress,<br/>uint8_t * pStatus)</b>                                                                                                                                                    |
| Function description | Erase sector from PCCARD memory.                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpccard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li> <li>• <b>SectorAddress:</b> Sector address to erase</li> <li>• <b>pStatus:</b> pointer to CF status</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                  |

**HAL\_PCCARD\_Reset**

|                      |                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCCARD_Reset<br/>(PCCARD_HandleTypeDef * hpcard)</b>                                                                                               |
| Function description | Reset the PCCARD memory.                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                        |

**HAL\_PCCARD\_IRQHandler**

|                      |                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCCARD_IRQHandler (PCCARD_HandleTypeDef * hpcard)</b>                                                                                                           |
| Function description | This function handles PCCARD device interrupt request.                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                        |

**HAL\_PCCARD\_ITCallback**

|                      |                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCCARD_ITCallback (PCCARD_HandleTypeDef * hpcard)</b>                                                                                                           |
| Function description | PCCARD interrupt feature callback.                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                               |

**HAL\_PCCARD\_GetState**

|                      |                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_PCCARD_StateTypeDef HAL_PCCARD_GetState<br/>(PCCARD_HandleTypeDef * hpcard)</b>                                                                                      |
| Function description | return the PCCARD controller state                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> state</li></ul>                                                                                                         |

**HAL\_PCCARD\_GetStatus**

|                      |                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_PCCARD_StatusTypeDef HAL_PCCARD_GetStatus<br/>(PCCARD_HandleTypeDef * hpccard)</b>                                                                                                                                                                                                                                           |
| Function description | Get the compact flash memory status.                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpccard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li> </ul>                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>New:</b> status of the CF operation. This parameter can be: <ul style="list-style-type: none"> <li>– CompactFlash_TIMEOUT_ERROR: when the previous operation generate a Timeout error</li> <li>– CompactFlash_READY: when memory is ready for the next operation</li> </ul> </li> </ul> |

**HAL\_PCCARD\_ReadStatus**

|                      |                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_PCCARD_StatusTypeDef HAL_PCCARD_ReadStatus<br/>(PCCARD_HandleTypeDef * hpccard)</b>                                                                                                                                                                                                                                                                                   |
| Function description | Reads the Compact Flash memory status using the Read status command.                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpccard:</b> pointer to a PCCARD_HandleTypeDef structure that contains the configuration information for PCCARD module.</li> </ul>                                                                                                                                                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>The:</b> status of the Compact Flash memory. This parameter can be: <ul style="list-style-type: none"> <li>– CompactFlash_BUSY: when memory is busy</li> <li>– CompactFlash_READY: when memory is ready for the next operation</li> <li>– CompactFlash_ERROR: when the previous operation generates error</li> </ul> </li> </ul> |

## 35.3 PCCARD Firmware driver defines

### 35.3.1 PCCARD

**PCCARD Exported Macros**

| <u>_HAL_PCCARD_RESET_HANDLE_STATE</u> | <u>Description:</u>                                                                               |
|---------------------------------------|---------------------------------------------------------------------------------------------------|
|                                       | <ul style="list-style-type: none"> <li>• Reset PCCARD handle state.</li> </ul>                    |
|                                       | <b>Parameters:</b>                                                                                |
|                                       | <ul style="list-style-type: none"> <li>• <u>_HANDLE_</u>: specifies the PCCARD handle.</li> </ul> |
|                                       | <b>Return value:</b>                                                                              |
|                                       | <ul style="list-style-type: none"> <li>• None</li> </ul>                                          |

## 36 HAL PCD Generic Driver

### 36.1 PCD Firmware driver registers structures

#### 36.1.1 PCD\_InitTypeDef

##### Data Fields

- *uint32\_t dev\_endpoints*
- *uint32\_t speed*
- *uint32\_t ep0\_mps*
- *uint32\_t phy\_itface*
- *uint32\_t Sof\_enable*
- *uint32\_t low\_power\_enable*
- *uint32\_t lpm\_enable*
- *uint32\_t battery\_charging\_enable*

##### Field Documentation

- ***uint32\_t PCD\_InitTypeDef::dev\_endpoints***  
Device Endpoints number. This parameter depends on the used USB core. This parameter must be a number between Min\_Data = 1 and Max\_Data = 15
- ***uint32\_t PCD\_InitTypeDef::speed***  
USB Core speed. This parameter can be any value of [PCD\\_Core\\_Speed](#)
- ***uint32\_t PCD\_InitTypeDef::ep0\_mps***  
Set the Endpoint 0 Max Packet size. This parameter can be any value of [PCD\\_EP0\\_MPS](#)
- ***uint32\_t PCD\_InitTypeDef::phy\_itface***  
Select the used PHY interface. This parameter can be any value of [PCD\\_Core\\_PHY](#)
- ***uint32\_t PCD\_InitTypeDef::Sof\_enable***  
Enable or disable the output of the SOF signal. This parameter can be set to ENABLE or DISABLE
- ***uint32\_t PCD\_InitTypeDef::low\_power\_enable***  
Enable or disable Low Power mode This parameter can be set to ENABLE or DISABLE
- ***uint32\_t PCD\_InitTypeDef::lpm\_enable***  
Enable or disable the Link Power Management . This parameter can be set to ENABLE or DISABLE
- ***uint32\_t PCD\_InitTypeDef::battery\_charging\_enable***  
Enable or disable Battery charging. This parameter can be set to ENABLE or DISABLE

#### 36.1.2 PCD\_EPTypeDef

##### Data Fields

- *uint8\_t num*
- *uint8\_t is\_in*
- *uint8\_t is\_stall*
- *uint8\_t type*
- *uint16\_t pmaaddress*
- *uint16\_t pmaaddr0*
- *uint16\_t pmaaddr1*
- *uint8\_t doublebuffer*

- *uint32\_t maxpacket*
- *uint8\_t \* xfer\_buff*
- *uint32\_t xfer\_len*
- *uint32\_t xfer\_count*

#### Field Documentation

- *uint8\_t PCD\_EPTypedef::num*  
Endpoint number This parameter must be a number between Min\_Data = 1 and Max\_Data = 15
- *uint8\_t PCD\_EPTypedef::is\_in*  
Endpoint direction This parameter must be a number between Min\_Data = 0 and Max\_Data = 1
- *uint8\_t PCD\_EPTypedef::is\_stall*  
Endpoint stall condition This parameter must be a number between Min\_Data = 0 and Max\_Data = 1
- *uint8\_t PCD\_EPTypedef::type*  
Endpoint type This parameter can be any value of [PCD\\_EP\\_Type](#)
- *uint16\_t PCD\_EPTypedef::pmaaddress*  
PMA Address This parameter can be any value between Min\_addr = 0 and Max\_addr = 1K
- *uint16\_t PCD\_EPTypedef::pmaaddr0*  
PMA Address0 This parameter can be any value between Min\_addr = 0 and Max\_addr = 1K
- *uint16\_t PCD\_EPTypedef::pmaaddr1*  
PMA Address1 This parameter can be any value between Min\_addr = 0 and Max\_addr = 1K
- *uint8\_t PCD\_EPTypedef::doublebuffer*  
Double buffer enable This parameter can be 0 or 1
- *uint32\_t PCD\_EPTypedef::maxpacket*  
Endpoint Max packet size This parameter must be a number between Min\_Data = 0 and Max\_Data = 64KB
- *uint8\_t\* PCD\_EPTypedef::xfer\_buff*  
Pointer to transfer buffer
- *uint32\_t PCD\_EPTypedef::xfer\_len*  
Current transfer length
- *uint32\_t PCD\_EPTypedef::xfer\_count*  
Partial transfer length in case of multi packet transfer

### 36.1.3 PCD\_HandleTypeDefDef

#### Data Fields

- *PCD\_TypeDef \* Instance*
- *PCD\_InitTypeDef Init*
- *\_IO uint8\_t USB\_Address*
- *PCD\_EPTypedef IN\_ep*
- *PCD\_EPTypedef OUT\_ep*
- *HAL\_LockTypeDef Lock*
- *\_IO PCD\_StateTypeDef State*
- *uint32\_t Setup*
- *void \* pData*

#### Field Documentation

- *PCD\_TypeDef\* PCD\_HandleTypeDefDef::Instance*  
Register base address

- ***PCD\_InitTypeDef PCD\_HandleTypeDef::Init***  
PCD required parameters
- ***\_IO uint8\_t PCD\_HandleTypeDef::USB\_Address***  
USB Address
- ***PCD\_EPTypeDef PCD\_HandleTypeDef::IN\_ep[15]***  
IN endpoint parameters
- ***PCD\_EPTypeDef PCD\_HandleTypeDef::OUT\_ep[15]***  
OUT endpoint parameters
- ***HAL\_LockTypeDef PCD\_HandleTypeDef::Lock***  
PCD peripheral status
- ***\_IO PCD\_StateTypeDef PCD\_HandleTypeDef::State***  
PCD communication state
- ***uint32\_t PCD\_HandleTypeDef::Setup[12]***  
Setup packet buffer
- ***void\* PCD\_HandleTypeDef::pData***  
Pointer to upper stack Handler

## 36.2 PCD Firmware driver API description

### 36.2.1 How to use this driver

The PCD HAL driver can be used as follows:

1. Declare a PCD\_HandleTypeDef handle structure, for example: PCD\_HandleTypeDef hpcd;
2. Fill parameters of Init structure in HCD handle
3. Call HAL\_PCD\_Init() API to initialize the HCD peripheral (Core, Device core, ...)
4. Initialize the PCD low level resources through the HAL\_PCD\_MspInit() API:
  - a. Enable the PCD/USB Low Level interface clock using
    - \_\_HAL\_RCC\_USB\_CLK\_ENABLE();
  - b. Initialize the related GPIO clocks
  - c. Configure PCD pin-out
  - d. Configure PCD NVIC interrupt
5. Associate the Upper USB device stack to the HAL PCD Driver:
  - a. hpcd.pData = pdev;
6. Enable HCD transmission and reception:
  - a. HAL\_PCD\_Start();

### 36.2.2 Initialization and de-initialization functions

This section provides functions allowing to:

This section contains the following APIs:

- [\*\*HAL\\_PCD\\_Init\(\)\*\*](#)
- [\*\*HAL\\_PCD\\_DelInit\(\)\*\*](#)
- [\*\*HAL\\_PCD\\_MspInit\(\)\*\*](#)
- [\*\*HAL\\_PCD\\_MspDelInit\(\)\*\*](#)

### 36.2.3 IO operation functions

This subsection provides a set of functions allowing to manage the PCD data transfers.

This section contains the following APIs:

- [\*\*HAL\\_PCD\\_Start\(\)\*\*](#)
- [\*\*HAL\\_PCD\\_Stop\(\)\*\*](#)
- [\*\*HAL\\_PCD\\_IRQHandler\(\)\*\*](#)

- [\*HAL\\_PCD\\_DataOutStageCallback\(\)\*](#)
- [\*HAL\\_PCD\\_DataInStageCallback\(\)\*](#)
- [\*HAL\\_PCD\\_SetupStageCallback\(\)\*](#)
- [\*HAL\\_PCD\\_SOFCallback\(\)\*](#)
- [\*HAL\\_PCD\\_ResetCallback\(\)\*](#)
- [\*HAL\\_PCD\\_SuspendCallback\(\)\*](#)
- [\*HAL\\_PCD\\_ResumeCallback\(\)\*](#)
- [\*HAL\\_PCD\\_ISOOUTIncompleteCallback\(\)\*](#)
- [\*HAL\\_PCD\\_ISOINIncompleteCallback\(\)\*](#)
- [\*HAL\\_PCD\\_ConnectCallback\(\)\*](#)
- [\*HAL\\_PCD\\_DisconnectCallback\(\)\*](#)

### 36.2.4 Peripheral Control functions

This subsection provides a set of functions allowing to control the PCD data transfers.

This section contains the following APIs:

- [\*HAL\\_PCD\\_DevConnect\(\)\*](#)
- [\*HAL\\_PCD\\_DevDisconnect\(\)\*](#)
- [\*HAL\\_PCD\\_SetAddress\(\)\*](#)
- [\*HAL\\_PCD\\_EP\\_Open\(\)\*](#)
- [\*HAL\\_PCD\\_EP\\_Close\(\)\*](#)
- [\*HAL\\_PCD\\_EP\\_Receive\(\)\*](#)
- [\*HAL\\_PCD\\_EP\\_GetRxCount\(\)\*](#)
- [\*HAL\\_PCD\\_EP\\_Transmit\(\)\*](#)
- [\*HAL\\_PCD\\_EP\\_SetStall\(\)\*](#)
- [\*HAL\\_PCD\\_EP\\_ClrStall\(\)\*](#)
- [\*HAL\\_PCD\\_EP\\_Flush\(\)\*](#)
- [\*HAL\\_PCD\\_ActivateRemoteWakeup\(\)\*](#)
- [\*HAL\\_PCD\\_DeActivateRemoteWakeup\(\)\*](#)
- [\*HAL\\_PCD\\_ActiveRemoteWakeup\(\)\*](#)
- [\*HAL\\_PCD\\_DeActiveRemoteWakeup\(\)\*](#)

### 36.2.5 Peripheral State functions

This subsection permits to get in run-time the status of the peripheral and the data flow.

This section contains the following APIs:

- [\*HAL\\_PCD\\_GetState\(\)\*](#)

### 36.2.6 Detailed description of functions

#### **HAL\_PCD\_Init**

|                      |                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_Init (PCD_HandleTypeDef * hpcd)</b>                                                   |
| Function description | Initializes the PCD according to the specified parameters in the PCD_InitTypeDef and create the associated handle. |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> </ul>                                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                             |

**HAL\_PCD\_DelInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_DelInit (PCD_HandleTypeDef * hpcd)</b>         |
| Function description | Deinitializes the PCD peripheral.                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>      |

**HAL\_PCD\_MspInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_MspInit (PCD_HandleTypeDef * hpcd)</b>                      |
| Function description | Initializes the PCD MSP.                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_PCD\_MspDeInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_MspDeInit (PCD_HandleTypeDef * hpcd)</b>                    |
| Function description | Deinitializes PCD MSP.                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_PCD\_Start**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_Start (PCD_HandleTypeDef * hpcd)</b>           |
| Function description | Start the USB device.                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>      |

**HAL\_PCD\_Stop**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_Stop (PCD_HandleTypeDef * hpcd)</b>            |
| Function description | Stop the USB device.                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>      |

**HAL\_PCD\_IRQHandler**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_IRQHandler (PCD_HandleTypeDef * hpcd)</b>                   |
| Function description | This function handles PCD interrupt request.                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>      |

**HAL\_PCD\_DataOutStageCallback**

|                      |                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_DataOutStageCallback (PCD_HandleTypeDef * hpcd, uint8_t epnum)</b>                                   |
| Function description | Data out stage callbacks.                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> <li>• <b>epnum:</b> endpoint number</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                      |

**HAL\_PCD\_DataInStageCallback**

|                      |                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_DataInStageCallback (PCD_HandleTypeDef * hpcd, uint8_t epnum)</b>                                    |
| Function description | Data IN stage callbacks.                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> <li>• <b>epnum:</b> endpoint number</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                      |

**HAL\_PCD\_SetupStageCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_SetupStageCallback (PCD_HandleTypeDef * hpcd)</b>           |
| Function description | Setup stage callback.                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_PCD\_SOFCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_SOFCallback (PCD_HandleTypeDef * hpcd)</b>                  |
| Function description | USB Start Of Frame callbacks.                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_PCD\_ResetCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_ResetCallback (PCD_HandleTypeDef * hpcd)</b>                |
| Function description | USB Reset callbacks.                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_PCD\_SuspendCallback**

|                      |                                                                |
|----------------------|----------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_SuspendCallback (PCD_HandleTypeDef * hpcd)</b> |
| Function description | Suspend event callbacks.                                       |

---

|               |                                                                           |
|---------------|---------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"><li>• <b>hpcd:</b> PCD handle</li></ul> |
| Return values | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>             |

### **HAL\_PCD\_ResumeCallback**

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_ResumeCallback (PCD_HandleTypeDef * hpcd)</b>             |
| Function description | Resume event callbacks.                                                   |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcd:</b> PCD handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>             |

### **HAL\_PCD\_ISOOUTIncompleteCallback**

|                      |                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_ISOOUTIncompleteCallback (PCD_HandleTypeDef * hpcd, uint8_t epi)</b>                            |
| Function description | Incomplete ISO OUT callbacks.                                                                                   |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcd:</b> PCD handle</li><li>• <b>epi:</b> endpoint number</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                   |

### **HAL\_PCD\_ISOINIncompleteCallback**

|                      |                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_ISOINIncompleteCallback (PCD_HandleTypeDef * hpcd, uint8_t epi)</b>                             |
| Function description | Incomplete ISO IN callbacks.                                                                                    |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcd:</b> PCD handle</li><li>• <b>epi:</b> endpoint number</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                   |

### **HAL\_PCD\_ConnectCallback**

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_ConnectCallback (PCD_HandleTypeDef * hpcd)</b>            |
| Function description | Connection event callbacks.                                               |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcd:</b> PCD handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>             |

### **HAL\_PCD\_DisconnectCallback**

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCD_DisconnectCallback (PCD_HandleTypeDef * hpcd)</b>         |
| Function description | Disconnection event callbacks.                                            |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcd:</b> PCD handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>             |

**HAL\_PCD\_DevConnect**

Function name      **HAL\_StatusTypeDef HAL\_PCD\_DevConnect  
(PCD\_HandleTypeDef \* hpcd)**

Function description      Connect the USB device.

Parameters      • **hpcd:** PCD handle

Return values      • **HAL:** status

**HAL\_PCD\_DevDisconnect**

Function name      **HAL\_StatusTypeDef HAL\_PCD\_DevDisconnect  
(PCD\_HandleTypeDef \* hpcd)**

Function description      Disconnect the USB device.

Parameters      • **hpcd:** PCD handle

Return values      • **HAL:** status

**HAL\_PCD\_SetAddress**

Function name      **HAL\_StatusTypeDef HAL\_PCD\_SetAddress  
(PCD\_HandleTypeDef \* hpcd, uint8\_t address)**

Function description      Set the USB Device address.

Parameters      • **hpcd:** PCD handle

• **address:** new device address

Return values      • **HAL:** status

**HAL\_PCD\_EP\_Open**

Function name      **HAL\_StatusTypeDef HAL\_PCD\_EP\_Open  
(PCD\_HandleTypeDef \* hpcd, uint8\_t ep\_addr, uint16\_t  
ep\_mps, uint8\_t ep\_type)**

Function description      Open and configure an endpoint.

Parameters      • **hpcd:** PCD handle

• **ep\_addr:** endpoint address

• **ep\_mps:** endpoint max packet size

• **ep\_type:** endpoint type

Return values      • **HAL:** status

**HAL\_PCD\_EP\_Close**

Function name      **HAL\_StatusTypeDef HAL\_PCD\_EP\_Close  
(PCD\_HandleTypeDef \* hpcd, uint8\_t ep\_addr)**

Function description      Deactivate an endpoint.

Parameters      • **hpcd:** PCD handle

• **ep\_addr:** endpoint address

Return values      • **HAL:** status

**HAL\_PCD\_EP\_Receive**

|                      |                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_EP_Receive<br/>(PCD_HandleTypeDef * hpcd, uint8_t ep_addr, uint8_t * pBuf,<br/>uint32_t len)</b>                                                                                                        |
| Function description | Receive an amount of data.                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> <li>• <b>ep_addr:</b> endpoint address</li> <li>• <b>pBuf:</b> pointer to the reception buffer</li> <li>• <b>len:</b> amount of data to be received</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                               |

**HAL\_PCD\_EP\_Transmit**

|                      |                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_EP_Transmit<br/>(PCD_HandleTypeDef * hpcd, uint8_t ep_addr, uint8_t * pBuf,<br/>uint32_t len)</b>                                                                                                      |
| Function description | Send an amount of data.                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> <li>• <b>ep_addr:</b> endpoint address</li> <li>• <b>pBuf:</b> pointer to the transmission buffer</li> <li>• <b>len:</b> amount of data to be sent</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                              |

**HAL\_PCD\_EP\_GetRxCount**

|                      |                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint16_t HAL_PCD_EP_GetRxCount (PCD_HandleTypeDef *<br/>hpcd, uint8_t ep_addr)</b>                                   |
| Function description | Get Received Data Size.                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> <li>• <b>ep_addr:</b> endpoint address</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Data:</b> Size</li> </ul>                                                   |

**HAL\_PCD\_EP\_SetStall**

|                      |                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_EP_SetStall<br/>(PCD_HandleTypeDef * hpcd, uint8_t ep_addr)</b>                            |
| Function description | Set a STALL condition over an endpoint.                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hpcd:</b> PCD handle</li> <li>• <b>ep_addr:</b> endpoint address</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                  |

**HAL\_PCD\_EP\_ClrStall**

|                      |                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_EP_ClrStall<br/>(PCD_HandleTypeDef * hpcd, uint8_t ep_addr)</b> |
| Function description | Clear a STALL condition over in an endpoint.                                                 |

---

|               |                                                                                                                     |
|---------------|---------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li><b>hpcd:</b> PCD handle</li> <li><b>ep_addr:</b> endpoint address</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                |

### HAL\_PCD\_EP\_Flush

|                      |                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_EP_Flush<br/>(PCD_HandleTypeDef * hpcd, uint8_t ep_addr)</b>                           |
| Function description | Flush an endpoint.                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>hpcd:</b> PCD handle</li> <li><b>ep_addr:</b> endpoint address</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                |

### HAL\_PCD\_ActivateRemoteWakeup

|                      |                                                                                      |
|----------------------|--------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_ActivateRemoteWakeup<br/>(PCD_HandleTypeDef * hpcd)</b> |
| Function description | HAL_PCD_ActivateRemoteWakeup : active remote wakeup signalling.                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>hpcd:</b> PCD handle</li> </ul>            |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                 |

### HAL\_PCD\_DeActivateRemoteWakeup

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_DeActivateRemoteWakeup<br/>(PCD_HandleTypeDef * hpcd)</b> |
| Function description | HAL_PCD_DeActivateRemoteWakeup : de-active remote wakeup signalling.                   |
| Parameters           | <ul style="list-style-type: none"> <li><b>hpcd:</b> PCD handle</li> </ul>              |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                   |

### HAL\_PCD\_GetState

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>PCD_StateTypeDef HAL_PCD_GetState (PCD_HandleTypeDef<br/>* hpcd)</b>   |
| Function description | Return the PCD state.                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>hpcd:</b> PCD handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> state</li> </ul>       |

### HAL\_PCD\_ActiveRemoteWakeup

|                      |                                                                                    |
|----------------------|------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCD_ActiveRemoteWakeup<br/>(PCD_HandleTypeDef * hpcd)</b> |
| Function description |                                                                                    |

**HAL\_PCD\_DeActiveRemoteWakeUp**

Function name      **HAL\_StatusTypeDef HAL\_PCD\_DeActiveRemoteWakeUp  
(PCD\_HandleTypeDef \* hpcd)**

Function description

**PCD\_WritePMA**

Function name      **void PCD\_WritePMA (USB\_TypeDef \* USBx, uint8\_t \* pbUsrBuf, uint16\_t wPMABufAddr, uint16\_t wNBytes)**

Function description      Copy a buffer from user memory area to packet memory area (PMA)

Parameters     

- **USBx:** USB peripheral instance register address.
- **pbUsrBuf:** pointer to user memory area.
- **wPMABufAddr:** address into PMA.
- **wNBytes:** no. of bytes to be copied.

Return values     

- **None**

**PCD\_ReadPMA**

Function name      **void PCD\_ReadPMA (USB\_TypeDef \* USBx, uint8\_t \* pbUsrBuf, uint16\_t wPMABufAddr, uint16\_t wNBytes)**

Function description      Copy a buffer from user memory area to packet memory area (PMA)

Parameters     

- **USBx:** USB peripheral instance register address.
- **pbUsrBuf:** pointer to user memory area.
- **wPMABufAddr:** address into PMA.
- **wNBytes:** no. of bytes to be copied.

Return values     

- **None**

## 36.3 PCD Firmware driver defines

### 36.3.1 PCD

*PCD Core PHY*

PCD\_PHY\_EMBEDDED

*PCD Core Speed*

PCD\_SPEED\_HIGH

PCD\_SPEED\_FULL

*PCD ENDP*

PCD\_ENDP0

PCD\_ENDP1

PCD\_ENDP2

PCD\_ENDP3

PCD\_ENDP4

PCD\_ENDP5

PCD\_ENDP6

PCD\_ENDP7

***PCD Endpoint Kind***

PCD\_SNG\_BUF

PCD\_DBL\_BUF

***PCD EP0 MPS***

DEP0CTL\_MPS\_64

DEP0CTL\_MPS\_32

DEP0CTL\_MPS\_16

DEP0CTL\_MPS\_8

PCD\_EP0MPS\_64

PCD\_EP0MPS\_32

PCD\_EP0MPS\_16

PCD\_EP0MPS\_08

***PCD EP Type***

PCD\_EP\_TYPE\_CTRL

PCD\_EP\_TYPE\_ISOC

PCD\_EP\_TYPE\_BULK

PCD\_EP\_TYPE\_INTR

***PCD Exported Macros***

\_HAL\_PCD\_GET\_FLAG

\_HAL\_PCD\_CLEAR\_FLAG

\_HAL\_USB\_WAKEUP\_EXTI\_ENABLE\_IT

\_HAL\_USB\_WAKEUP\_EXTI\_DISABLE\_IT

\_HAL\_USB\_EXTI\_GENERATE\_SWIT

\_HAL\_USB\_WAKEUP\_EXTI\_GET\_FLAG

\_HAL\_USB\_WAKEUP\_EXTI\_CLEAR\_FLAG

\_HAL\_USB\_WAKEUP\_EXTI\_ENABLE\_RISING\_EDGE

\_HAL\_USB\_WAKEUP\_EXTI\_ENABLE\_FALLING\_EDGE

\_HAL\_USB\_WAKEUP\_EXTI\_ENABLE\_RISING\_FALLING\_EDGE

***PCD Instance definition***

IS\_PCD\_ALL\_INSTANCE

## 37 HAL PCD Extension Driver

### 37.1 PCDEEx Firmware driver API description

#### 37.1.1 Extended Peripheral Control functions

This section provides functions allowing to:

- Update PMA configuration

This section contains the following APIs:

- [\*HAL\\_PCDEEx\\_PMACConfig\(\)\*](#)
- [\*HAL\\_PCDEEx\\_SetConnectionState\(\)\*](#)

#### 37.1.2 Detailed description of functions

##### **HAL\_PCDEEx\_PMACConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_PCDEEx_PMACConfig<br/>(PCD_HandleTypeDef * hpcd, uint16_t ep_addr, uint16_t<br/>ep_kind, uint32_t pmaaddress)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Configure PMA for EP.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcd:</b> PCD handle</li><li>• <b>ep_addr:</b> endpoint address</li><li>• <b>ep_kind:</b> endpoint Kind<ul style="list-style-type: none"><li>– USB_SNG_BUF: Single Buffer used</li><li>– USB_DBL_BUF: Double Buffer used</li></ul></li><li>• <b>pmaaddress:</b> EP address in The PMA: In case of single buffer endpoint this parameter is 16-bit value providing the address in PMA allocated to endpoint. In case of double buffer endpoint this parameter is a 32-bit value providing the endpoint buffer 0 address in the LSB part of 32-bit value and endpoint buffer 1 address in the MSB part of 32-bit value.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• :: status</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

##### **HAL\_PCDEEx\_SetConnectionState**

|                      |                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PCDEEx_SetConnectionState (PCD_HandleTypeDef *<br/>hpcd, uint8_t state)</b>                        |
| Function description | Software Device Connection.                                                                                    |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hpcd:</b> PCD handle</li><li>• <b>state:</b> Device state</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                  |

## 37.2 PCDEEx Firmware driver defines

### 37.2.1 PCDEEx

#### *PCD Extended Exported Macros*

`PCD_EP_TX_ADDRESS` **Description:**

- Gets address in an endpoint register.

**Parameters:**

- USBx: USB peripheral instance register address.
- bEpNum: Endpoint Number.

**Return value:**

- None

`PCD_EP_TX_CNT`

`PCD_EP_RX_ADDRESS`

`PCD_EP_RX_CNT`

`PCD_SET_EP_RX_CNT`

## 38 HAL PWR Generic Driver

### 38.1 PWR Firmware driver API description

#### 38.1.1 Initialization and de-initialization functions

After reset, the backup domain (RTC registers, RTC backup data registers and backup SRAM) is protected against possible unwanted write accesses. To enable access to the RTC Domain and RTC registers, proceed as follows:

- Enable the Power Controller (PWR) APB1 interface clock using the `__HAL_RCC_PWR_CLK_ENABLE()` macro.
- Enable access to RTC domain using the `HAL_PWR_EnableBkUpAccess()` function.

#### 38.1.2 Peripheral Control functions

##### WakeUp pin configuration

- WakeUp pin is used to wakeup the system from Standby mode. This pin is forced in input pull down configuration and is active on rising edges.
- There are up to three WakeUp pins:
  - WakeUp Pin 1 on PA.00.
  - WakeUp Pin 2 on PC.13 (STM32F303xC, STM32F303xE only).
  - WakeUp Pin 3 on PE.06.

##### Main and Backup Regulators configuration

- When the backup domain is supplied by VDD (analog switch connected to VDD) the backup SRAM is powered from VDD which replaces the VBAT power supply to save battery life.
- The backup SRAM is not mass erased by a tamper event. It is read protected to prevent confidential data, such as cryptographic private key, from being accessed. The backup SRAM can be erased only through the Flash interface when a protection level change from level 1 to level 0 is requested. Refer to the description of Read protection (RDP) in the Flash programming manual. Refer to the datasheets for more details.

##### Low Power modes configuration

The devices feature 3 low-power modes:

- Sleep mode: Cortex-M4 core stopped, peripherals kept running.
- Stop mode: all clocks are stopped, regulator running, regulator in low power mode
- Standby mode: 1.2V domain powered off (mode not available on STM32F3x8 devices).

##### Sleep mode

- Entry: The Sleep mode is entered by using the `HAL_PWR_EnterSLEEPMode(PWR_MAINREGULATOR_ON, PWR_SLEEPENTRY_WFx)` functions with
  - `PWR_SLEEPENTRY_WFI`: enter SLEEP mode with WFI instruction
  - `PWR_SLEEPENTRY_WFE`: enter SLEEP mode with WFE instruction
- Exit:

- Any peripheral interrupt acknowledged by the nested vectored interrupt controller (NVIC) can wake up the device from Sleep mode.

### Stop mode

In Stop mode, all clocks in the 1.8V domain are stopped, the PLL, the HSI, and the HSE RC oscillators are disabled. Internal SRAM and register contents are preserved. The voltage regulator can be configured either in normal or low-power mode to minimize the consumption.

- Entry: The Stop mode is entered using the HAL\_PWR\_EnterSTOPMode(PWR\_MAINREGULATOR\_ON, PWR\_STOPENTRY\_WFI ) function with:
  - Main regulator ON or
  - Low Power regulator ON.
  - PWR\_STOPENTRY\_WFI: enter STOP mode with WFI instruction or
  - PWR\_STOPENTRY\_WFE: enter STOP mode with WFE instruction
- Exit:
  - Any EXTI Line (Internal or External) configured in Interrupt/Event mode.
  - Some specific communication peripherals (CEC, USART, I2C) interrupts, when programmed in wakeup mode (the peripheral must be programmed in wakeup mode and the corresponding interrupt vector must be enabled in the NVIC).

### Standby mode

The Standby mode allows to achieve the lowest power consumption. It is based on the Cortex-M4 deep sleep mode, with the voltage regulator disabled. The 1.8V domain is consequently powered off. The PLL, the HSI oscillator and the HSE oscillator are also switched off. SRAM and register contents are lost except for the RTC registers, RTC backup registers, backup SRAM and Standby circuitry. The voltage regulator is OFF.

- Entry:
  - The Standby mode is entered using the HAL\_PWR\_EnterSTANDBYMode() function.
- Exit:
  - WKUP pin rising edge, RTC alarm (Alarm A and Alarm B), RTC wakeup, tamper event, time-stamp event, external reset in NRST pin, IWDG reset.

### Auto-wakeup (AWU) from low-power mode

The MCU can be woken up from low-power mode by an RTC Alarm event, an RTC Wakeup event, a tamper event, a time-stamp event, or a comparator event, without depending on an external interrupt (Auto-wakeup mode).

- RTC auto-wakeup (AWU) from the Stop and Standby modes
  - To wake up from the Stop mode with an RTC alarm event, it is necessary to configure the RTC to generate the RTC alarm using the HAL\_RTC\_SetAlarm\_IT() function.
  - To wake up from the Stop mode with an RTC Tamper or time stamp event, it is necessary to configure the RTC to detect the tamper or time stamp event using the HAL\_RTC\_SetTimeStamp\_IT() or HAL\_RTC\_SetTamper\_IT() functions.
  - To wake up from the Stop mode with an RTC WakeUp event, it is necessary to configure the RTC to generate the RTC WakeUp event using the HAL\_RTC\_SetWakeUpTimer\_IT() function.

- Comparator auto-wakeup (AWU) from the Stop mode
  - To wake up from the Stop mode with a comparator wakeup event, it is necessary to:
    - Configure the EXTI Line associated with the comparator (example EXTI Line 22 for comparator 2U) to be sensitive to the selected edges (falling, rising or falling and rising) (Interrupt or Event modes) using the EXTI\_Init() function.
    - Configure the comparator to generate the event.

This section contains the following APIs:

- [\*\*\*HAL\\_PWR\\_EnableWakeUpPin\(\)\*\*\*](#)
- [\*\*\*HAL\\_PWR\\_DisableWakeUpPin\(\)\*\*\*](#)
- [\*\*\*HAL\\_PWR\\_EnterSLEEPMode\(\)\*\*\*](#)
- [\*\*\*HAL\\_PWR\\_EnterSTOPMode\(\)\*\*\*](#)
- [\*\*\*HAL\\_PWR\\_EnterSTANDBYMode\(\)\*\*\*](#)
- [\*\*\*HAL\\_PWR\\_EnableSleepOnExit\(\)\*\*\*](#)
- [\*\*\*HAL\\_PWR\\_DisableSleepOnExit\(\)\*\*\*](#)
- [\*\*\*HAL\\_PWR\\_EnableSEVOnPend\(\)\*\*\*](#)
- [\*\*\*HAL\\_PWR\\_DisableSEVOnPend\(\)\*\*\*](#)
- [\*\*\*HAL\\_PWR\\_EnableBkUpAccess\(\)\*\*\*](#)
- [\*\*\*HAL\\_PWR\\_DisableBkUpAccess\(\)\*\*\*](#)

### 38.1.3 Detailed description of functions

#### ***HAL\_PWR\_DeInit***

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b><i>void HAL_PWR_DeInit (void )</i></b>                                 |
| Function description | Deinitializes the PWR peripheral registers to their default reset values. |
| Return values        | <ul style="list-style-type: none"> <li>• <b><i>None</i></b></li> </ul>    |

#### ***HAL\_PWR\_EnableBkUpAccess***

|                      |                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><i>void HAL_PWR_EnableBkUpAccess (void )</i></b>                                                                                                     |
| Function description | Enables access to the backup domain (RTC registers, RTC backup data registers and backup SRAM).                                                         |
| Return values        | <ul style="list-style-type: none"> <li>• <b><i>None</i></b></li> </ul>                                                                                  |
| Notes                | <ul style="list-style-type: none"> <li>• If the HSE divided by 32 is used as the RTC clock, the Backup Domain Access should be kept enabled.</li> </ul> |

#### ***HAL\_PWR\_DisableBkUpAccess***

|                      |                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><i>void HAL_PWR_DisableBkUpAccess (void )</i></b>                                                                                                    |
| Function description | Disables access to the backup domain (RTC registers, RTC backup data registers and backup SRAM).                                                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b><i>None</i></b></li> </ul>                                                                                  |
| Notes                | <ul style="list-style-type: none"> <li>• If the HSE divided by 32 is used as the RTC clock, the Backup Domain Access should be kept enabled.</li> </ul> |

**HAL\_PWR\_EnableWakeUpPin**

|                      |                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_EnableWakeUpPin (uint32_t WakeUpPinx)</b>                                                                                                              |
| Function description | Enables the WakeUp PINx functionality.                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>WakeUpPinx:</b> Specifies the Power Wake-Up pin to enable.<br/>This parameter can be value of : PWR WakeUp Pins</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                        |

**HAL\_PWR\_DisableWakeUpPin**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_DisableWakeUpPin (uint32_t WakeUpPinx)</b>                                                                                                               |
| Function description | Disables the WakeUp PINx functionality.                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>WakeUpPinx:</b> Specifies the Power Wake-Up pin to disable.<br/>This parameter can be values of : PWR WakeUp Pins</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                          |

**HAL\_PWR\_EnterSTOPMode**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_EnterSTOPMode (uint32_t Regulator, uint8_t STOPEntry)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description | Enters STOP mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Regulator:</b> Specifies the regulator state in STOP mode. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– PWR_MAINREGULATOR_ON: STOP mode with regulator ON</li> <li>– PWR_LOWPOWERREGULATOR_ON: STOP mode with low power regulator ON</li> </ul> </li> <li>• <b>STOPEntry:</b> specifies if STOP mode is entered with WFI or WFE instruction. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– PWR_STOPENTRY_WFI: Enter STOP mode with WFI instruction</li> <li>– PWR_STOPENTRY_WFE: Enter STOP mode with WFE instruction</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• In Stop mode, all I/O pins keep the same state as in Run mode.</li> <li>• When exiting Stop mode by issuing an interrupt or a wakeup event, the HSI RC oscillator is selected as system clock.</li> <li>• When the voltage regulator operates in low power mode, an additional startup delay is incurred when waking up from Stop mode. By keeping the internal regulator ON during Stop mode, the consumption is higher although the startup time is reduced.</li> </ul>                                                                                                                                                                    |

**HAL\_PWR\_EnterSLEEPMode**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_EnterSLEEPMode (uint32_t Regulator, uint8_t SLEEPEntry)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Enters Sleep mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Regulator:</b> Specifies the regulator state in SLEEP mode. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– PWR_MAINREGULATOR_ON: SLEEP mode with regulator ON</li> <li>– PWR_LOWPOWERREGULATOR_ON: SLEEP mode with low power regulator ON</li> </ul> </li> <li>• <b>SLEEPEntry:</b> Specifies if SLEEP mode is entered with WFI or WFE instruction. When WFI entry is used, tick interrupt have to be disabled if not desired as the interrupt wake up source. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– PWR_SLEEPENTRY_WFI: enter SLEEP mode with WFI instruction</li> <li>– PWR_SLEEPENTRY_WFE: enter SLEEP mode with WFE instruction</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• In Sleep mode, all I/O pins keep the same state as in Run mode.</li> <li>• This parameter has no effect in F3 family and is just maintained to offer full portability of other STM32 families softwares.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

**HAL\_PWR\_EnterSTANDBYMode**

|                      |                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_EnterSTANDBYMode (void )</b>                                                                                                                                                                                                                                       |
| Function description | Enters STANDBY mode.                                                                                                                                                                                                                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• In Standby mode, all I/O pins are high impedance except for: Reset pad (still available), RTC alternate function pins if configured for tamper, time-stamp, RTC Alarm out, or RTC clock calibration out, WKUP pins if enabled.</li> </ul> |

**HAL\_PWR\_EnableSleepOnExit**

|                      |                                                                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_EnableSleepOnExit (void )</b>                                                                                                                                                                                                                                             |
| Function description | Indicates Sleep-On-Exit when returning from Handler mode to Thread mode.                                                                                                                                                                                                                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• Set SLEEPONEXIT bit of SCR register. When this bit is set, the processor re-enters SLEEP mode when an interruption handling is over. Setting this bit is useful when the processor is expected to run only on interruptions handling.</li> </ul> |

**HAL\_PWR\_DisableSleepOnExit**

|                      |                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_DisableSleepOnExit (void )</b>                                                                                                                                              |
| Function description | Disables Sleep-On-Exit feature when returning from Handler mode to Thread mode.                                                                                                             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• Clears SLEEPONEXIT bit of SCR register. When this bit is set, the processor re-enters SLEEP mode when an interruption handling is over.</li> </ul> |

**HAL\_PWR\_EnableSEVOnPend**

|                      |                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_EnableSEVOnPend (void )</b>                                                                                                                                               |
| Function description | Enables CORTEX M4 SEVONPEND bit.                                                                                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• Sets SEVONPEND bit of SCR register. When this bit is set, this causes WFE to wake up when an interrupt moves from inactive to pended.</li> </ul> |

**HAL\_PWR\_DisableSEVOnPend**

|                      |                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_DisableSEVOnPend (void )</b>                                                                                                                                                |
| Function description | Disables CORTEX M4 SEVONPEND bit.                                                                                                                                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• Clears SEVONPEND bit of SCR register. When this bit is set, this causes WFE to wake up when an interrupt moves from inactive to pended.</li> </ul> |

## 38.2 PWR Firmware driver defines

### 38.2.1 PWR

*PWR Exported Macro*

| <u>_HAL_PWR_GET_FLAG</u> | <b>Description:</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|--------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                          | <ul style="list-style-type: none"> <li>• Check PWR flag is set or not.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|                          | <b>Parameters:</b> <ul style="list-style-type: none"> <li>• <u>_FLAG_</u>: specifies the flag to check. This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- PWR_FLAG_WU: Wake Up flag. This flag indicates that a wakeup event was received from the WKUP pin or from the RTC alarm (Alarm A or Alarm B), RTC Tamper event, RTC TimeStamp event or RTC Wakeup. An additional wakeup event is detected if the WKUP pin is enabled (by setting the EWUP bit) when the WKUP pin level is already high.</li> <li>- PWR_FLAG_SB: StandBy flag. This flag indicates that the system was resumed from</li> </ul> </li> </ul> |

StandBy mode.

- PWR\_FLAG\_PVDO: PVD Output. This flag is valid only if PVD is enabled by the HAL\_PWR\_EnablePVD() function. The PVD is stopped by Standby mode. For this reason, this bit is equal to 0 after Standby or reset until the PVDE bit is set.
- PWR\_FLAG\_VREFINTRDY: This flag indicates that the internal reference voltage VREFINT is ready.

#### **Return value:**

- The new state of \_\_FLAG\_\_ (TRUE or FALSE).

### \_\_HAL\_PWR\_CLEAR\_FLAG

#### **Description:**

- Clear the PWR's pending flags.

#### **Parameters:**

- \_\_FLAG\_\_: specifies the flag to clear. This parameter can be one of the following values:
  - PWR\_FLAG\_WU: Wake Up flag
  - PWR\_FLAG\_SB: StandBy flag

#### **PWR Flag**

|                     |                                           |
|---------------------|-------------------------------------------|
| PWR_FLAG_WU         | Wakeup event from wakeup pin or RTC alarm |
| PWR_FLAG_SB         | Standby flag                              |
| PWR_FLAG_PVDO       | Power Voltage Detector output flag        |
| PWR_FLAG_VREFINTRDY | VREFINT reference voltage ready           |

#### **PWR Regulator state in STOP mode**

|                          |                                                      |
|--------------------------|------------------------------------------------------|
| PWR_MAINREGULATOR_ON     | Voltage regulator on during STOP mode                |
| PWR_LOWPOWERREGULATOR_ON | Voltage regulator in low-power mode during STOP mode |

#### **PWR SLEEP mode entry**

|                    |                                                       |
|--------------------|-------------------------------------------------------|
| PWR_SLEEPENTRY_WFI | Wait For Interruption instruction to enter SLEEP mode |
| PWR_SLEEPENTRY_WFE | Wait For Event instruction to enter SLEEP mode        |

#### **PWR STOP mode entry**

|                   |                                                      |
|-------------------|------------------------------------------------------|
| PWR_STOPENTRY_WFI | Wait For Interruption instruction to enter STOP mode |
| PWR_STOPENTRY_WFE | Wait For Event instruction to enter STOP mode        |

#### **PWR WakeUp Pins**

|                 |               |
|-----------------|---------------|
| PWR_WAKEUP_PIN1 | Wakeup pin 1U |
| PWR_WAKEUP_PIN2 | Wakeup pin 2U |
| PWR_WAKEUP_PIN3 | Wakeup pin 3U |

## 39 HAL PWR Extension Driver

### 39.1 PWREx Firmware driver registers structures

#### 39.1.1 PWR\_PVDTTypeDef

##### Data Fields

- *uint32\_t PVDLevel*
- *uint32\_t Mode*

##### Field Documentation

- *uint32\_t PWR\_PVDTTypeDef::PVDLevel*

PVDLevel: Specifies the PVD detection level This parameter can be a value of [\*\*PWREx\\_PVD\\_detection\\_level\*\*](#)

- *uint32\_t PWR\_PVDTTypeDef::Mode*

Mode: Specifies the operating mode for the selected pins. This parameter can be a value of [\*\*PWREx\\_PVD\\_Mode\*\*](#)

### 39.2 PWREx Firmware driver API description

#### 39.2.1 Peripheral Extended control functions

##### PVD configuration (present on all other devices than STM32F3x8 devices)

- The PVD is used to monitor the VDD power supply by comparing it to a threshold selected by the PVD Level (PLS[2:0] bits in the PWR\_CR).
- A PVDO flag is available to indicate if VDD/VDDA is higher or lower than the PVD threshold. This event is internally connected to the EXTI line16 and can generate an interrupt if enabled. This is done through `__HAL_PWR_PVD_EXTI_ENABLE_IT()` macro
- The PVD is stopped in Standby mode. PVD is not available on STM32F3x8 Product Line

##### Voltage regulator

- The voltage regulator is always enabled after Reset. It works in three different modes. In Run mode, the regulator supplies full power to the 1.8V domain (core, memories and digital peripherals). In Stop mode, the regulator supplies low power to the 1.8V domain, preserving contents of registers and SRAM. In Stop mode, the regulator is powered off. The contents of the registers and SRAM are lost except for the Standby circuitry and the Backup Domain. Note: in the STM32F3x8xx devices, the voltage regulator is bypassed and the microcontroller must be powered from a nominal VDD = 1.8V +/-8U% voltage.
- A PVDO flag is available to indicate if VDD/VDDA is higher or lower than the PVD threshold. This event is internally connected to the EXTI line16 and can generate an interrupt if enabled. This is done through `__HAL_PWR_PVD_EXTI_ENABLE_IT()` macro
- The PVD is stopped in Standby mode.

### SDADC power configuration

- On STM32F373xC/STM32F378xx devices, there are up to 3 SDADC instances that can be enabled/disabled.

This section contains the following APIs:

- [\*HAL\\_PWR\\_ConfigPVD\(\)\*](#)
- [\*HAL\\_PWR\\_EnablePVD\(\)\*](#)
- [\*HAL\\_PWR\\_DisablePVD\(\)\*](#)
- [\*HAL\\_PWR\\_PVD\\_IRQHandler\(\)\*](#)
- [\*HAL\\_PWR\\_PVDCallback\(\)\*](#)

## 39.2.2 Detailed description of functions

### HAL\_PWR\_ConfigPVD

|                      |                                                                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_ConfigPVD (PWR_PVDTTypeDef * sConfigPVD)</b>                                                                                                                                           |
| Function description | Configures the voltage threshold detected by the Power Voltage Detector(PVD).                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"><li>• <b>sConfigPVD</b>: pointer to an PWR_PVDTTypeDef structure that contains the configuration information for the PVD.</li></ul>                                  |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"><li>• Refer to the electrical characteristics of your device datasheet for more details about the voltage threshold corresponding to each detection level.</li></ul> |

### HAL\_PWR\_EnablePVD

|                      |                                                               |
|----------------------|---------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_EnablePVD (void )</b>                         |
| Function description | Enables the Power Voltage Detector(PVD).                      |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul> |

### HAL\_PWR\_DisablePVD

|                      |                                                               |
|----------------------|---------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_DisablePVD (void )</b>                        |
| Function description | Disables the Power Voltage Detector(PVD).                     |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul> |

### HAL\_PWR\_PVD\_IRQHandler

|                      |                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_PWR_PVD_IRQHandler (void )</b>                                                              |
| Function description | This function handles the PWR PVD interrupt request.                                                    |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                           |
| Notes                | <ul style="list-style-type: none"><li>• This API should be called under the PVD_IRQHandler().</li></ul> |

**HAL\_PWR\_PVDCallback**

Function name      **void HAL\_PWR\_PVDCallback (void )**

Function description      PWR PVD interrupt callback.

Return values      •    **None**

## 39.3 PWREx Firmware driver defines

### 39.3.1 PWREx

#### *PWR Extended Exported Constants*

**PWR\_EXTI\_LINE\_PVD**   External interrupt line 16 Connected to the PVD EXTI Line

#### *PWR Extended Exported Macros*

|                                                    |                                                                                                                  |
|----------------------------------------------------|------------------------------------------------------------------------------------------------------------------|
| <code>_HAL_PWR_PVD_EXTI_ENABLE_IT</code>           | <b>Description:</b><br>• Enable interrupt on PVD Exti Line 16.                                                   |
| <code>_HAL_PWR_PVD_EXTI_DISABLE_IT</code>          | <b>Return value:</b><br>• None.<br><b>Description:</b><br>• Disable interrupt on PVD Exti Line 16.               |
| <code>_HAL_PWR_PVD_EXTI_GENERATE_SWIT</code>       | <b>Return value:</b><br>• None.<br><b>Description:</b><br>• Generate a Software interrupt on selected EXTI line. |
| <code>_HAL_PWR_PVD_EXTI_ENABLE_EVENT</code>        | <b>Return value:</b><br>• None.<br><b>Description:</b><br>• Enable event on PVD Exti Line 16.                    |
| <code>_HAL_PWR_PVD_EXTI_DISABLE_EVENT</code>       | <b>Return value:</b><br>• None.<br><b>Description:</b><br>• Disable event on PVD Exti Line 16.                   |
| <code>_HAL_PWR_PVD_EXTI_DISABLE_RISING_EDGE</code> | <b>Description:</b>                                                                                              |

- Disable the PVD Extended Interrupt Rising Trigger.

**Return value:**

- None.

`__HAL_PWR_PVD_EXTI_DISABLE_FALLING_EDGE`

- Disable the PVD Extended Interrupt Falling Trigger.

**Return value:**

- None.

`__HAL_PWR_PVD_EXTI_DISABLE_RISING_FALLING_EDGE`

**Description:**

- Disable the PVD Extended Interrupt Rising & Falling Trigger.

**Return value:**

- None

`__HAL_PWR_PVD_EXTI_ENABLE_FALLING_EDGE`

**Description:**

- PVD EXTI line configuration: set falling edge trigger.

**Return value:**

- None.

`__HAL_PWR_PVD_EXTI_ENABLE_RISING_EDGE`

**Description:**

- PVD EXTI line configuration: set rising edge trigger.

**Return value:**

- None.

`__HAL_PWR_PVD_EXTI_ENABLE_RISING_FALLING_EDGE`

**Description:**

- Enable the PVD Extended Interrupt Rising & Falling Trigger.

**Return value:**

- None

`__HAL_PWR_PVD_EXTI_GET_FLAG`

**Description:**

- Check whether the specified PVD EXTI interrupt flag is set or not.

**Return value:**

- EXTI: PVD Line Status.

`__HAL_PWR_PVD_EXTI_CLEAR_FLAG`

**Description:**

- Clear the PVD EXTI flag.

**Return value:**

- None.

**PWR Extended PVD detection level**

|                |                            |
|----------------|----------------------------|
| PWR_PVDLEVEL_0 | PVD threshold around 2.2 V |
| PWR_PVDLEVEL_1 | PVD threshold around 2.3 V |
| PWR_PVDLEVEL_2 | PVD threshold around 2.4 V |
| PWR_PVDLEVEL_3 | PVD threshold around 2.5 V |
| PWR_PVDLEVEL_4 | PVD threshold around 2.6 V |
| PWR_PVDLEVEL_5 | PVD threshold around 2.7 V |
| PWR_PVDLEVEL_6 | PVD threshold around 2.8 V |
| PWR_PVDLEVEL_7 | PVD threshold around 2.9 V |

**PWR Extended PVD Mode**

|                                   |                                                                    |
|-----------------------------------|--------------------------------------------------------------------|
| PWR_PVD_MODE_NORMAL               | Basic mode is used                                                 |
| PWR_PVD_MODE_IT_RISING            | External Interrupt Mode with Rising edge trigger detection         |
| PWR_PVD_MODE_IT_FALLING           | External Interrupt Mode with Falling edge trigger detection        |
| PWR_PVD_MODE_IT_RISING_FALLING    | External Interrupt Mode with Rising/Falling edge trigger detection |
| PWR_PVD_MODE_EVENT_RISING         | Event Mode with Rising edge trigger detection                      |
| PWR_PVD_MODE_EVENT_FALLING        | Event Mode with Falling edge trigger detection                     |
| PWR_PVD_MODE_EVENT_RISING_FALLING | Event Mode with Rising/Falling edge trigger detection              |

## 40 HAL RCC Generic Driver

### 40.1 RCC Firmware driver registers structures

#### 40.1.1 RCC\_PLLInitTypeDef

##### Data Fields

- *uint32\_t PLLState*
- *uint32\_t PLLSource*
- *uint32\_t PLLMUL*

##### Field Documentation

- *uint32\_t RCC\_PLLInitTypeDef::PLLState*  
PLLState: The new state of the PLL. This parameter can be a value of [RCC\\_PLL\\_Config](#)
- *uint32\_t RCC\_PLLInitTypeDef::PLLSource*  
PLLSource: PLL entry clock source. This parameter must be a value of [RCC\\_PLL\\_Clock\\_Source](#)
- *uint32\_t RCC\_PLLInitTypeDef::PLLMUL*  
PLLMUL: Multiplication factor for PLL VCO input clock This parameter must be a value of [RCC\\_PLL\\_Multiplication\\_Factor](#)

#### 40.1.2 RCC\_OsclInitTypeDef

##### Data Fields

- *uint32\_t OscillatorType*
- *uint32\_t HSEState*
- *uint32\_t HSEPredivValue*
- *uint32\_t LSEState*
- *uint32\_t HSISState*
- *uint32\_t HSICalibrationValue*
- *uint32\_t LSISState*
- *RCC\_PLLInitTypeDef PLL*

##### Field Documentation

- *uint32\_t RCC\_OsclInitTypeDef::OscillatorType*  
The oscillators to be configured. This parameter can be a value of [RCC\\_Oscillator\\_Type](#)
- *uint32\_t RCC\_OsclInitTypeDef::HSEState*  
The new state of the HSE. This parameter can be a value of [RCC\\_HSE\\_Config](#)
- *uint32\_t RCC\_OsclInitTypeDef::HSEPredivValue*  
The HSE predivision factor value. This parameter can be a value of [RCC\\_PLL\\_HSE\\_Prediv\\_Factor](#)
- *uint32\_t RCC\_OsclInitTypeDef::LSEState*  
The new state of the LSE. This parameter can be a value of [RCC\\_LSE\\_Config](#)
- *uint32\_t RCC\_OsclInitTypeDef::HSISState*  
The new state of the HSI. This parameter can be a value of [RCC\\_HSI\\_Config](#)
- *uint32\_t RCC\_OsclInitTypeDef::HSICalibrationValue*  
The HSI calibration trimming value (default is RCC\_HSICALIBRATION\_DEFAULT).  
This parameter must be a number between Min\_Data = 0x00 and Max\_Data = 0x1FU

- *uint32\_t RCC\_OscInitTypeDef::LSIState*  
The new state of the LSI. This parameter can be a value of [RCC\\_LSI\\_Config](#)
- *RCC\_PLLInitTypeDef RCC\_OscInitTypeDef::PLL*  
PLL structure parameters

### 40.1.3 RCC\_ClkInitTypeDef

#### Data Fields

- *uint32\_t ClockType*
- *uint32\_t SYSCLKSource*
- *uint32\_t AHBCLKDivider*
- *uint32\_t APB1CLKDivider*
- *uint32\_t APB2CLKDivider*

#### Field Documentation

- *uint32\_t RCC\_ClkInitTypeDef::ClockType*  
The clock to be configured. This parameter can be a value of [RCC\\_System\\_Clock\\_Type](#)
- *uint32\_t RCC\_ClkInitTypeDef::SYSCLKSource*  
The clock source (SYSCLK) used as system clock. This parameter can be a value of [RCC\\_System\\_Clock\\_Source](#)
- *uint32\_t RCC\_ClkInitTypeDef::AHBCLKDivider*  
The AHB clock (HCLK) divider. This clock is derived from the system clock (SYSCLK).  
This parameter can be a value of [RCC\\_AHB\\_Clock\\_Source](#)
- *uint32\_t RCC\_ClkInitTypeDef::APB1CLKDivider*  
The APB1 clock (PCLK1) divider. This clock is derived from the AHB clock (HCLK).  
This parameter can be a value of [RCC\\_APB1\\_APB2\\_Clock\\_Source](#)
- *uint32\_t RCC\_ClkInitTypeDef::APB2CLKDivider*  
The APB2 clock (PCLK2) divider. This clock is derived from the AHB clock (HCLK).  
This parameter can be a value of [RCC\\_APB1\\_APB2\\_Clock\\_Source](#)

## 40.2 RCC Firmware driver API description

### 40.2.1 RCC specific features

After reset the device is running from Internal High Speed oscillator (HSI 8MHz) with Flash 0 wait state, Flash prefetch buffer is enabled, and all peripherals are off except internal SRAM, Flash and JTAG.

- There is no prescaler on High speed (AHB) and Low speed (APB) buses; all peripherals mapped on these buses are running at HSI speed.
- The clock for all peripherals is switched off, except the SRAM and FLASH.
- All GPIOs are in input floating state, except the JTAG pins which are assigned to be used for debug purpose.

Once the device started from reset, the user application has to:

- Configure the clock source to be used to drive the System clock (if the application needs higher frequency/performance)
- Configure the System clock frequency and Flash settings
- Configure the AHB and APB buses prescalers
- Enable the clock for the peripheral(s) to be used
- Configure the clock source(s) for peripherals whose clocks are not derived from the System clock (RTC, ADC, I2C, I2S, TIM, USB FS)

## 40.2.2 RCC Limitations

A delay between an RCC peripheral clock enable and the effective peripheral enabling should be taken into account in order to manage the peripheral read/write from/to registers.

- This delay depends on the peripheral mapping.
  - AHB & APB peripherals, 1 dummy read is necessary

Workarounds:

1. For AHB & APB peripherals, a dummy read to the peripheral register has been inserted in each `__HAL_RCC_PPP_CLK_ENABLE()` macro.

## 40.2.3 Initialization and de-initialization functions

This section provides functions allowing to configure the internal/external oscillators (HSE, HSI, LSE, LSI, PLL, CSS and MCO) and the System buses clocks (SYSCLK, AHB, APB1 and APB2).

Internal/external clock and PLL configuration

1. HSI (high-speed internal), 8 MHz factory-trimmed RC used directly or through the PLL as System clock source. The HSI clock can be used also to clock the USART and I2C peripherals.
2. LSI (low-speed internal), ~40 KHz low consumption RC used as IWDG and/or RTC clock source.
3. HSE (high-speed external), 4 to 32 MHz crystal oscillator used directly or through the PLL as System clock source. Can be used also as RTC clock source.
4. LSE (low-speed external), 32 KHz oscillator used as RTC clock source.
5. PLL (clocked by HSI or HSE), featuring different output clocks:
  - The first output is used to generate the high speed system clock (up to 72 MHz)
  - The second output is used to generate the clock for the USB FS (48 MHz)
  - The third output may be used to generate the clock for the ADC peripherals (up to 72 MHz)
  - The fourth output may be used to generate the clock for the TIM peripherals (144 MHz)
6. CSS (Clock security system), once enable using the macro `__HAL_RCC_CSS_ENABLE()` and if a HSE clock failure occurs(HSE used directly or through PLL as System clock source), the System clocks automatically switched to HSI and an interrupt is generated if enabled. The interrupt is linked to the Cortex-M4 NMI (Non-Maskable Interrupt) exception vector.
7. MCO (microcontroller clock output), used to output SYSCLK, HSI, HSE, LSI, LSE or PLL clock (divided by 2) output on pin (such as PA8 pin).

System, AHB and APB buses clocks configuration

1. Several clock sources can be used to drive the System clock (SYSCLK): HSI, HSE and PLL. The AHB clock (HCLK) is derived from System clock through configurable prescaler and used to clock the CPU, memory and peripherals mapped on AHB bus (DMA, GPIO...). APB1 (PCLK1) and APB2 (PCLK2) clocks are derived from AHB clock through configurable prescalers and used to clock the peripherals mapped on these buses. You can use "`@ref HAL_RCC_GetSysClockFreq()`" function to retrieve the frequencies of these clocks.
2. All the peripheral clocks are derived from the System clock (SYSCLK) except:
  - The FLASH program/erase clock which is always HSI 8MHz clock.
  - The USB 48 MHz clock which is derived from the PLL VCO clock.
  - The USART clock which can be derived as well from HSI 8MHz, LSI or LSE.
  - The I2C clock which can be derived as well from HSI 8MHz clock.
  - The ADC clock which is derived from PLL output.

- The RTC clock which is derived from the LSE, LSI or 1 MHz HSE\_RTC (HSE divided by a programmable prescaler). The System clock (SYSCLK) frequency must be higher or equal to the RTC clock frequency.
  - IWDG clock which is always the LSI clock.
3. For the STM32F3xx devices, the maximum frequency of the SYSCLK, HCLK, PCLK1 and PCLK2 is 72 MHz, Depending on the SYSCLK frequency, the flash latency should be adapted accordingly.
  4. After reset, the System clock source is the HSI (8 MHz) with 0 WS and prefetch is disabled.

This section contains the following APIs:

- [\*HAL\\_RCC\\_DelInit\(\)\*](#)
- [\*HAL\\_RCC\\_OscConfig\(\)\*](#)
- [\*HAL\\_RCC\\_ClockConfig\(\)\*](#)

#### 40.2.4 Peripheral Control functions

This subsection provides a set of functions allowing to control the RCC Clocks frequencies.

This section contains the following APIs:

- [\*HAL\\_RCC\\_MCOConfig\(\)\*](#)
- [\*HAL\\_RCC\\_EnableCSS\(\)\*](#)
- [\*HAL\\_RCC\\_DisableCSS\(\)\*](#)
- [\*HAL\\_RCC\\_GetSysClockFreq\(\)\*](#)
- [\*HAL\\_RCC\\_GetHCLKFreq\(\)\*](#)
- [\*HAL\\_RCC\\_GetPCLK1Freq\(\)\*](#)
- [\*HAL\\_RCC\\_GetPCLK2Freq\(\)\*](#)
- [\*HAL\\_RCC\\_GetOscConfig\(\)\*](#)
- [\*HAL\\_RCC\\_GetClockConfig\(\)\*](#)
- [\*HAL\\_RCC\\_NMI\\_IRQHandler\(\)\*](#)
- [\*HAL\\_RCC\\_CSSCallback\(\)\*](#)

#### 40.2.5 Detailed description of functions

##### **HAL\_RCC\_DelInit**

|                      |                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_RCC_DelInit (void )</b>                                                                                                                                                                                                                                                                                                                                  |
| Function description | Resets the RCC clock configuration to the default reset state.                                                                                                                                                                                                                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• The default reset state of the clock configuration is given below: HSI ON and used as system clock sourceHSE and PLL OFFAHB, APB1 and APB2 prescaler set to 1.CSS and MCO1 OFFAll interrupts disabled</li> <li>• This function does not modify the configuration of the Peripheral clocksLSI, LSE and RTC clocks</li> </ul> |

##### **HAL\_RCC\_OscConfig**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RCC_OscConfig<br/>(RCC_OscInitTypeDef * RCC_OscInitStruct)</b> |
| Function description | Initializes the RCC Oscillators according to the specified                              |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | parameters in the RCC_OscInitTypeDef.                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters    | <ul style="list-style-type: none"> <li><b>RCC_OscInitStruct:</b> pointer to an RCC_OscInitTypeDef structure that contains the configuration information for the RCC Oscillators.</li> </ul>                                                                                                                                                                                                                                                                                   |
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                          |
| Notes         | <ul style="list-style-type: none"> <li>The PLL is not disabled when used as system clock.</li> <li>Transitions LSE Bypass to LSE On and LSE On to LSE Bypass are not supported by this macro. User should request a transition to LSE Off first and then LSE On or LSE Bypass.</li> <li>Transition HSE Bypass to HSE On and HSE On to HSE Bypass are not supported by this macro. User should request a transition to HSE Off first and then HSE On or HSE Bypass.</li> </ul> |

### HAL\_RCC\_ClockConfig

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RCC_ClockConfig (RCC_ClkInitTypeDef * RCC_ClkInitStruct, uint32_t FLatency)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Initializes the CPU, AHB and APB buses clocks according to the specified parameters in the RCC_ClkInitStruct.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>RCC_ClkInitStruct:</b> pointer to an RCC_OscInitTypeDef structure that contains the configuration information for the RCC peripheral.</li> <li><b>FLatency:</b> FLASH Latency The value of this parameter depend on device used within the same series</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>The SystemCoreClock CMSIS variable is used to store System Clock Frequency and updated by HAL_RCC_GetHCLKFreq() function called within this function</li> <li>The HSI is used (enabled by hardware) as system clock source after start-up from Reset, wake-up from STOP and STANDBY mode, or in case of failure of the HSE used directly or indirectly as system clock (if the Clock Security System CSS is enabled).</li> <li>A switch from one clock source to another occurs only if the target clock source is ready (clock stable after start-up delay or PLL locked). If a clock source which is not yet ready is selected, the switch will occur when the clock source will be ready. You can use HAL_RCC_GetClockConfig() function to know which clock is currently used as system clock source.</li> </ul> |

### HAL\_RCC\_MCOConfig

|                      |                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_RCC_MCOConfig (uint32_t RCC_MCOx, uint32_t RCC_MCOsource, uint32_t RCC_MCODiv)</b>                                                                                                                                                                                                                                                                                                          |
| Function description | Selects the clock source to output on MCO pin.                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>RCC_MCOx:</b> specifies the output direction for the clock source. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>RCC_MCO1 Clock source to output on MCO1 pin(PA8).</li> </ul> </li> <li><b>RCC_MCOsource:</b> specifies the clock source to output. This parameter can be one of the following values:</li> </ul> |

---

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>- RCC_MCO1SOURCE_NOCLOCK No clock selected as MCO clock</li> <li>- RCC_MCO1SOURCE_SYSCLK System clock selected as MCO clock</li> <li>- RCC_MCO1SOURCE_HSI HSI selected as MCO clock</li> <li>- RCC_MCO1SOURCE_HSE HSE selected as MCO clock</li> <li>- RCC_MCO1SOURCE_LSI LSI selected as MCO clock</li> <li>- RCC_MCO1SOURCE_LSE LSE selected as MCO clock</li> <li>- RCC_MCO1SOURCE_PLLCLK_DIV2 PLLCLK Divided by 2 selected as MCO clock</li> </ul> <ul style="list-style-type: none"> <li>• <b>RCC_MCODiv:</b> specifies the MCO DIV. This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- RCC_MCODIV_1 no division applied to MCO clock</li> </ul> </li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Notes         | <ul style="list-style-type: none"> <li>• MCO pin should be configured in alternate function mode.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

### HAL\_RCC\_EnableCSS

|                      |                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_RCC_EnableCSS (void )</b>                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Enables the Clock Security System.                                                                                                                                                                                                                                                                                                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• If a failure is detected on the HSE oscillator clock, this oscillator is automatically disabled and an interrupt is generated to inform the software about the failure (Clock Security System Interrupt, CSSI), allowing the MCU to perform rescue operations. The CSSI is linked to the Cortex-M4 NMI (Non-Maskable Interrupt) exception vector.</li> </ul> |

### HAL\_RCC\_NMI\_IRQHandler

|                      |                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_RCC_NMI_IRQHandler (void )</b>                                                             |
| Function description | This function handles the RCC CSS interrupt request.                                                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                        |
| Notes                | <ul style="list-style-type: none"> <li>• This API should be called under the NMI_Handler().</li> </ul> |

### HAL\_RCC\_CSSCallback

|                      |                                                                 |
|----------------------|-----------------------------------------------------------------|
| Function name        | <b>void HAL_RCC_CSSCallback (void )</b>                         |
| Function description | RCC Clock Security System interrupt callback.                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |

### HAL\_RCC\_DisableCSS

|                      |                                                                 |
|----------------------|-----------------------------------------------------------------|
| Function name        | <b>void HAL_RCC_DisableCSS (void )</b>                          |
| Function description | Disables the Clock Security System.                             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |

**HAL\_RCC\_GetSysClockFreq**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_RCC_GetSysClockFreq (void )</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description | Returns the SYSCLK frequency.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"><li><b>SYSCLK:</b> frequency</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"><li>The system frequency computed by this function is not the real frequency in the chip. It is calculated based on the predefined constant and the selected clock source:</li><li>If SYSCLK source is HSI, function returns values based on HSI_VALUE(*)</li><li>If SYSCLK source is HSE, function returns a value based on HSE_VALUE divided by PREDIV factor(**)</li><li>If SYSCLK source is PLL, function returns a value based on HSE_VALUE divided by PREDIV factor(**) or HSI_VALUE(*) multiplied by the PLL factor.</li><li>(*) HSI_VALUE is a constant defined in stm32f3xx_hal_conf.h file (default value 8 MHz) but the real value may vary depending on the variations in voltage and temperature.</li><li>(**) HSE_VALUE is a constant defined in stm32f3xx_hal_conf.h file (default value 8 MHz), user has to ensure that HSE_VALUE is same as the real frequency of the crystal used. Otherwise, this function may have wrong result.</li><li>The result of this function could be not correct when using fractional value for HSE crystal.</li><li>This function can be used by the user application to compute the baud-rate for the communication peripherals or configure other parameters.</li><li>Each time SYSCLK changes, this function must be called to update the right SYSCLK value. Otherwise, any configuration based on this function will be incorrect.</li></ul> |

**HAL\_RCC\_GetHCLKFreq**

|                      |                                                                                                                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_RCC_GetHCLKFreq (void )</b>                                                                                                                                                                                                                                                                                     |
| Function description | Returns the HCLK frequency.                                                                                                                                                                                                                                                                                                     |
| Return values        | <ul style="list-style-type: none"><li><b>HCLK:</b> frequency</li></ul>                                                                                                                                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"><li>Each time HCLK changes, this function must be called to update the right HCLK value. Otherwise, any configuration based on this function will be incorrect.</li><li>The SystemCoreClock CMSIS variable is used to store System Clock Frequency and updated within this function</li></ul> |

**HAL\_RCC\_GetPCLK1Freq**

|                      |                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_RCC_GetPCLK1Freq (void )</code>                                                                                                                                                                |
| Function description | Returns the PCLK1 frequency.                                                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>PCLK1:</b> frequency</li> </ul>                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• Each time PCLK1 changes, this function must be called to update the right PCLK1 value. Otherwise, any configuration based on this function will be incorrect.</li> </ul> |

**HAL\_RCC\_GetPCLK2Freq**

|                      |                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_RCC_GetPCLK2Freq (void )</code>                                                                                                                                                                |
| Function description | Returns the PCLK2 frequency.                                                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>PCLK2:</b> frequency</li> </ul>                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• Each time PCLK2 changes, this function must be called to update the right PCLK2 value. Otherwise, any configuration based on this function will be incorrect.</li> </ul> |

**HAL\_RCC\_GetOscConfig**

|                      |                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void HAL_RCC_GetOscConfig (RCC_OscInitTypeDef * RCC_OscInitStruct)</code>                                                                   |
| Function description | Configures the RCC_OscInitStruct according to the internal RCC configuration registers.                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RCC_OscInitStruct:</b> pointer to an RCC_OscInitTypeDef structure that will be configured.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                   |

**HAL\_RCC\_GetClockConfig**

|                      |                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void HAL_RCC_GetClockConfig (RCC_ClkInitTypeDef * RCC_ClkInitStruct, uint32_t * pFLatency)</code>                                                                                                                            |
| Function description | Get the RCC_ClkInitStruct according to the internal RCC configuration registers.                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RCC_ClkInitStruct:</b> pointer to an RCC_ClkInitTypeDef structure that contains the current clock configuration.</li> <li>• <b>pFLatency:</b> Pointer on the Flash Latency.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                    |

## 40.3 RCC Firmware driver defines

### 40.3.1 RCC

#### *RCC AHB Clock Enable Disable*

`_HAL_RCC_GPIOA_CLK_ENABLE`  
`_HAL_RCC_GPIOB_CLK_ENABLE`

```
_HAL_RCC_GPIOC_CLK_ENABLE
_HAL_RCC_GPIOD_CLK_ENABLE
_HAL_RCC_GPIOF_CLK_ENABLE
_HAL_RCC_CRC_CLK_ENABLE
_HAL_RCC_DMA1_CLK_ENABLE
_HAL_RCC_SRAM_CLK_ENABLE
_HAL_RCC_FLITF_CLK_ENABLE
_HAL_RCC_TSC_CLK_ENABLE
_HAL_RCC_GPIOA_CLK_DISABLE
_HAL_RCC_GPIOB_CLK_DISABLE
_HAL_RCC_GPIOC_CLK_DISABLE
_HAL_RCC_GPIOD_CLK_DISABLE
_HAL_RCC_GPIOF_CLK_DISABLE
_HAL_RCC_CRC_CLK_DISABLE
_HAL_RCC_DMA1_CLK_DISABLE
_HAL_RCC_SRAM_CLK_DISABLE
_HAL_RCC_FLITF_CLK_DISABLE
_HAL_RCC_TSC_CLK_DISABLE
```

**AHB Clock Source**

|                   |                       |
|-------------------|-----------------------|
| RCC_SYSCLK_DIV1   | SYSCLK not divided    |
| RCC_SYSCLK_DIV2   | SYSCLK divided by 2   |
| RCC_SYSCLK_DIV4   | SYSCLK divided by 4   |
| RCC_SYSCLK_DIV8   | SYSCLK divided by 8   |
| RCC_SYSCLK_DIV16  | SYSCLK divided by 16  |
| RCC_SYSCLK_DIV64  | SYSCLK divided by 64  |
| RCC_SYSCLK_DIV128 | SYSCLK divided by 128 |
| RCC_SYSCLK_DIV256 | SYSCLK divided by 256 |
| RCC_SYSCLK_DIV512 | SYSCLK divided by 512 |

**RCC AHB Force Release Reset**

```
_HAL_RCC_AHB_FORCE_RESET
_HAL_RCC_GPIOA_FORCE_RESET
_HAL_RCC_GPIOB_FORCE_RESET
_HAL_RCC_GPIOC_FORCE_RESET
_HAL_RCC_GPIOD_FORCE_RESET
_HAL_RCC_GPIOF_FORCE_RESET
_HAL_RCC_TSC_FORCE_RESET
```

\_HAL\_RCC\_AHB\_RELEASE\_RESET  
\_HAL\_RCC\_GPIOA\_RELEASE\_RESET  
\_HAL\_RCC\_GPIOB\_RELEASE\_RESET  
\_HAL\_RCC\_GPIOC\_RELEASE\_RESET  
\_HAL\_RCC\_GPIOD\_RELEASE\_RESET  
\_HAL\_RCC\_GPIOF\_RELEASE\_RESET  
\_HAL\_RCC\_TSC\_RELEASE\_RESET

**AHB Peripheral Clock Enable Disable Status**

\_HAL\_RCC\_GPIOA\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_GPIOB\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_GPIOC\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_GPIOD\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_GPIOF\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_CRC\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_DMA1\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_SRAM\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_FLITF\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_TSC\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_GPIOA\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_GPIOB\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_GPIOC\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_GPIOD\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_GPIOF\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_CRC\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_DMA1\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_SRAM\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_FLITF\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_TSC\_IS\_CLK\_DISABLED

**APB1 APB2 Clock Source**

|                |                    |
|----------------|--------------------|
| RCC_HCLK_DIV1  | HCLK not divided   |
| RCC_HCLK_DIV2  | HCLK divided by 2  |
| RCC_HCLK_DIV4  | HCLK divided by 4  |
| RCC_HCLK_DIV8  | HCLK divided by 8  |
| RCC_HCLK_DIV16 | HCLK divided by 16 |

**RCC APB1 Clock Enable Disable**

\_HAL\_RCC\_TIM2\_CLK\_ENABLE

```
_HAL_RCC_TIM6_CLK_ENABLE
_HAL_RCC_WWDG_CLK_ENABLE
_HAL_RCC_USART2_CLK_ENABLE
_HAL_RCC_USART3_CLK_ENABLE
_HAL_RCC_I2C1_CLK_ENABLE
_HAL_RCC_PWR_CLK_ENABLE
_HAL_RCC_DAC1_CLK_ENABLE
_HAL_RCC_TIM2_CLK_DISABLE
_HAL_RCC_TIM6_CLK_DISABLE
_HAL_RCC_WWDG_CLK_DISABLE
_HAL_RCC_USART2_CLK_DISABLE
_HAL_RCC_USART3_CLK_DISABLE
_HAL_RCC_I2C1_CLK_DISABLE
_HAL_RCC_PWR_CLK_DISABLE
_HAL_RCC_DAC1_CLK_DISABLE
```

***APB1 Peripheral Clock Enable Disable Status***

```
_HAL_RCC_TIM2_IS_CLK_ENABLED
_HAL_RCC_TIM6_IS_CLK_ENABLED
_HAL_RCC_WWDG_IS_CLK_ENABLED
_HAL_RCC_USART2_IS_CLK_ENABLED
_HAL_RCC_USART3_IS_CLK_ENABLED
_HAL_RCC_I2C1_IS_CLK_ENABLED
_HAL_RCC_PWR_IS_CLK_ENABLED
_HAL_RCC_DAC1_IS_CLK_ENABLED
_HAL_RCC_TIM2_IS_CLK_DISABLED
_HAL_RCC_TIM6_IS_CLK_DISABLED
_HAL_RCC_WWDG_IS_CLK_DISABLED
_HAL_RCC_USART2_IS_CLK_DISABLED
_HAL_RCC_USART3_IS_CLK_DISABLED
_HAL_RCC_I2C1_IS_CLK_DISABLED
_HAL_RCC_PWR_IS_CLK_DISABLED
_HAL_RCC_DAC1_IS_CLK_DISABLED
```

***RCC APB1 Force Release Reset***

```
_HAL_RCC_APB1_FORCE_RESET
_HAL_RCC_TIM2_FORCE_RESET
_HAL_RCC_TIM6_FORCE_RESET
```

```
_HAL_RCC_WWDG_FORCE_RESET
_HAL_RCC_USART2_FORCE_RESET
_HAL_RCC_USART3_FORCE_RESET
_HAL_RCC_I2C1_FORCE_RESET
_HAL_RCC_PWR_FORCE_RESET
_HAL_RCC_DAC1_FORCE_RESET
_HAL_RCC_APB1_RELEASE_RESET
_HAL_RCC_TIM2_RELEASE_RESET
_HAL_RCC_TIM6_RELEASE_RESET
_HAL_RCC_WWDG_RELEASE_RESET
_HAL_RCC_USART2_RELEASE_RESET
_HAL_RCC_USART3_RELEASE_RESET
_HAL_RCC_I2C1_RELEASE_RESET
_HAL_RCC_PWR_RELEASE_RESET
_HAL_RCC_DAC1_RELEASE_RESET
```

**RCC APB2 Clock Enable Disable**

```
_HAL_RCC_SYSCFG_CLK_ENABLE
_HAL_RCC_TIM15_CLK_ENABLE
_HAL_RCC_TIM16_CLK_ENABLE
_HAL_RCC_TIM17_CLK_ENABLE
_HAL_RCC_USART1_CLK_ENABLE
_HAL_RCC_SYSCFG_CLK_DISABLE
_HAL_RCC_TIM15_CLK_DISABLE
_HAL_RCC_TIM16_CLK_DISABLE
_HAL_RCC_TIM17_CLK_DISABLE
_HAL_RCC_USART1_CLK_DISABLE
```

**APB2 Peripheral Clock Enable Disable Status**

```
_HAL_RCC_SYSCFG_IS_CLK_ENABLED
_HAL_RCC_TIM15_IS_CLK_ENABLED
_HAL_RCC_TIM16_IS_CLK_ENABLED
_HAL_RCC_TIM17_IS_CLK_ENABLED
_HAL_RCC_USART1_IS_CLK_ENABLED
_HAL_RCC_SYSCFG_IS_CLK_DISABLED
_HAL_RCC_TIM15_IS_CLK_DISABLED
_HAL_RCC_TIM16_IS_CLK_DISABLED
_HAL_RCC_TIM17_IS_CLK_DISABLED
```

\_HAL\_RCC\_USART1\_IS\_CLK\_DISABLED  
**RCC APB2 Force Release Reset**  
\_HAL\_RCC\_APB2\_FORCE\_RESET  
\_HAL\_RCC\_SYSCFG\_FORCE\_RESET  
\_HAL\_RCC\_TIM15\_FORCE\_RESET  
\_HAL\_RCC\_TIM16\_FORCE\_RESET  
\_HAL\_RCC\_TIM17\_FORCE\_RESET  
\_HAL\_RCC\_USART1\_FORCE\_RESET  
\_HAL\_RCC\_APB2\_RELEASE\_RESET  
\_HAL\_RCC\_SYSCFG\_RELEASE\_RESET  
\_HAL\_RCC\_TIM15\_RELEASE\_RESET  
\_HAL\_RCC\_TIM16\_RELEASE\_RESET  
\_HAL\_RCC\_TIM17\_RELEASE\_RESET  
\_HAL\_RCC\_USART1\_RELEASE\_RESET

**BitAddress AliasRegion**

RCC\_CR\_OFFSET\_BB  
RCC\_CFGR\_OFFSET\_BB  
RCC\_CIR\_OFFSET\_BB  
RCC\_BDCR\_OFFSET\_BB  
RCC\_CSR\_OFFSET\_BB  
RCC\_HSION\_BIT\_NUMBER  
RCC\_CR\_HSION\_BB  
RCC\_HSEON\_BIT\_NUMBER  
RCC\_CR\_HSEON\_BB  
RCC\_CSSON\_BIT\_NUMBER  
RCC\_CR\_CSSON\_BB  
RCC\_PLLON\_BIT\_NUMBER  
RCC\_CR\_PLLON\_BB  
RCC\_LSION\_BIT\_NUMBER  
RCC\_CSR\_LSION\_BB  
RCC\_RMVF\_BIT\_NUMBER  
RCC\_CSR\_RMVF\_BB  
RCC\_LSEON\_BIT\_NUMBER  
RCC\_BDCR\_LSEON\_BB  
RCC\_LSEBYP\_BIT\_NUMBER  
RCC\_BDCR\_LSEBYP\_BB

RCC\_RTCEN\_BIT\_NUMBER  
 RCC\_BDCR\_RTCEN\_BB  
 RCC\_BDRST\_BIT\_NUMBER  
 RCC\_BDCR\_BDRST\_BB

**Flags**

|                    |                                      |
|--------------------|--------------------------------------|
| RCC_FLAG_HSIRDY    | Internal High Speed clock ready flag |
| RCC_FLAG_HSERDY    | External High Speed clock ready flag |
| RCC_FLAG_PLLRDY    | PLL clock ready flag                 |
| RCC_FLAG_LSIRDY    | Internal Low Speed oscillator Ready  |
| RCC_FLAG_V18PWRRST |                                      |
| RCC_FLAG_OBLRST    | Options bytes loading reset flag     |
| RCC_FLAG_PINRST    | PIN reset flag                       |
| RCC_FLAG_PORRST    | POR/PDR reset flag                   |
| RCC_FLAG_SFTRST    | Software Reset flag                  |
| RCC_FLAG_IWDGRST   | Independent Watchdog reset flag      |
| RCC_FLAG_WWDGRST   | Window watchdog reset flag           |
| RCC_FLAG_LPWRRST   | Low-Power reset flag                 |
| RCC_FLAG_LSERDY    | External Low Speed oscillator Ready  |
| RCC_FLAG_MCO       | Microcontroller Clock Output Flag    |

**Flags Interrupts Management**\_\_HAL\_RCC\_ENABLE\_IT**Description:**

- Enable RCC interrupt.

**Parameters:**

- \_\_INTERRUPT\_\_: specifies the RCC interrupt sources to be enabled. This parameter can be any combination of the following values:
  - RCC\_IT\_LSIRDY LSI ready interrupt
  - RCC\_IT\_LSERDY LSE ready interrupt
  - RCC\_IT\_HSIRDY HSI ready interrupt
  - RCC\_IT\_HSERDY HSE ready interrupt
  - RCC\_IT\_PLLRDY main PLL ready interrupt

\_\_HAL\_RCC\_DISABLE\_IT**Description:**

- Disable RCC interrupt.

**Parameters:**

- \_\_INTERRUPT\_\_: specifies the RCC interrupt sources to be disabled. This parameter can be any combination of the following values:
  - RCC\_IT\_LSIRDY LSI ready interrupt

- RCC\_IT\_LSERDY LSE ready interrupt
- RCC\_IT\_HSIRDY HSI ready interrupt
- RCC\_IT\_HSERDY HSE ready interrupt
- RCC\_IT\_PLLRDY main PLL ready interrupt

[\\_\\_HAL\\_RCC\\_CLEAR\\_IT](#)**Description:**

- Clear the RCC's interrupt pending bits.

**Parameters:**

- \_\_INTERRUPT\_\_: specifies the interrupt pending bit to clear. This parameter can be any combination of the following values:
  - RCC\_IT\_LSIRDY LSI ready interrupt.
  - RCC\_IT\_LSERDY LSE ready interrupt.
  - RCC\_IT\_HSIRDY HSI ready interrupt.
  - RCC\_IT\_HSERDY HSE ready interrupt.
  - RCC\_IT\_PLLRDY Main PLL ready interrupt.
  - RCC\_IT\_CSS Clock Security System interrupt

[\\_\\_HAL\\_RCC\\_GET\\_IT](#)**Description:**

- Check the RCC's interrupt has occurred or not.

**Parameters:**

- \_\_INTERRUPT\_\_: specifies the RCC interrupt source to check. This parameter can be one of the following values:
  - RCC\_IT\_LSIRDY LSI ready interrupt.
  - RCC\_IT\_LSERDY LSE ready interrupt.
  - RCC\_IT\_HSIRDY HSI ready interrupt.
  - RCC\_IT\_HSERDY HSE ready interrupt.
  - RCC\_IT\_PLLRDY Main PLL ready interrupt.
  - RCC\_IT\_CSS Clock Security System interrupt

**Return value:**

- The: new state of \_\_INTERRUPT\_\_ (TRUE or FALSE).

[\\_\\_HAL\\_RCC\\_CLEAR\\_RESET\\_FLAGS](#)

The reset flags are RCC\_FLAG\_PINRST, RCC\_FLAG\_PORRST, RCC\_FLAG\_SFTRST, RCC\_FLAG\_OBLRST, RCC\_FLAG\_IWDGRST, RCC\_FLAG\_WWDGRST, RCC\_FLAG\_LPWRRST

[\\_\\_HAL\\_RCC\\_GET\\_FLAG](#)**Description:**

- Check RCC flag is set or not.

**Parameters:**

- \_\_FLAG\_\_: specifies the flag to check. This parameter can be one of the following values:
  - RCC\_FLAG\_HSIRDY HSI oscillator clock ready.
  - RCC\_FLAG\_HSERDY HSE oscillator clock ready.
  - RCC\_FLAG\_PLLRDY Main PLL clock ready.
  - RCC\_FLAG\_LSERDY LSE oscillator clock ready.
  - RCC\_FLAG\_LSIRDY LSI oscillator clock ready.
  - RCC\_FLAG\_OBLRST Option Byte Load reset
  - RCC\_FLAG\_PINRST Pin reset.
  - RCC\_FLAG\_PORRST POR/PDR reset.
  - RCC\_FLAG\_SFTRST Software reset.
  - RCC\_FLAG\_IWDGRST Independent Watchdog reset.
  - RCC\_FLAG\_WWDGRST Window Watchdog reset.
  - RCC\_FLAG\_LPWRRST Low Power reset.

**Return value:**

- The new state of \_\_FLAG\_\_ (TRUE or FALSE).

**Get Clock source**\_HAL\_RCC\_SYSCLK\_CONFIG**Description:**

- Macro to configure the system clock source.

**Parameters:**

- \_\_SYSCLKSOURCE\_\_: specifies the system clock source. This parameter can be one of the following values:
  - RCC\_SYSCLKSOURCE\_HSI HSI oscillator is used as system clock source.
  - RCC\_SYSCLKSOURCE\_HSE HSE oscillator is used as system clock source.
  - RCC\_SYSCLKSOURCE\_PLLCLK PLL output is used as system clock source.

\_HAL\_RCC\_GET\_SYSCLK\_SOURCE**Description:**

- Macro to get the clock source used as system clock.

**Return value:**

- The clock source used as system clock. The returned value can be one of the following:
  - RCC\_SYSCLKSOURCE\_STATUS\_HSI HSI

- used as system clock
- RCC\_SYSCLKSOURCE\_STATUS\_HSE HSE used as system clock
- RCC\_SYSCLKSOURCE\_STATUS\_PLLCLK PLL used as system clock

#### **HSE Config**

|                |                                     |
|----------------|-------------------------------------|
| RCC_HSE_OFF    | HSE clock deactivation              |
| RCC_HSE_ON     | HSE clock activation                |
| RCC_HSE_BYPASS | External clock source for HSE clock |

#### **HSE Configuration**

##### \_\_HAL\_RCC\_HSE\_CONFIG    **Description:**

- Macro to configure the External High Speed oscillator (HSE).

##### **Parameters:**

- \_\_STATE\_\_: specifies the new state of the HSE. This parameter can be one of the following values:
  - RCC\_HSE\_OFF turn OFF the HSE oscillator, HSERDY flag goes low after 6 HSE oscillator clock cycles.
  - RCC\_HSE\_ON turn ON the HSE oscillator
  - RCC\_HSE\_BYPASS HSE oscillator bypassed with external clock

##### **Notes:**

- Transition HSE Bypass to HSE On and HSE On to HSE Bypass are not supported by this macro. User should request a transition to HSE Off first and then HSE On or HSE Bypass. After enabling the HSE (RCC\_HSE\_ON or RCC\_HSE\_Bypass), the application software should wait on HSERDY flag to be set indicating that HSE clock is stable and can be used to clock the PLL and/or system clock. HSE state can not be changed if it is used directly or through the PLL as system clock. In this case, you have to select another source of the system clock then change the HSE state (ex. disable it). The HSE is stopped by hardware when entering STOP and STANDBY modes. This function reset the CSSON bit, so if the clock security system(CSS) was previously enabled you have to enable it again after calling this function.

#### **HSI Config**

|                                   |                        |
|-----------------------------------|------------------------|
| RCC_HSI_OFF                       | HSI clock deactivation |
| RCC_HSI_ON                        | HSI clock activation   |
| <b>RCC_HSICALIBRATION_DEFAULT</b> |                        |

#### **HSI Configuration**

##### \_\_HAL\_RCC\_HSI\_ENABLE

##### **Notes:**

- The HSI is stopped by hardware when

entering STOP and STANDBY modes. It is used (enabled by hardware) as system clock source after startup from Reset, wakeup from STOP and STANDBY mode, or in case of failure of the HSE used directly or indirectly as system clock (if the Clock Security System CSS is enabled). HSI can not be stopped if it is used as system clock source. In this case, you have to select another source of the system clock then stop the HSI. After enabling the HSI, the application software should wait on HSIRDY flag to be set indicating that HSI clock is stable and can be used as system clock source. When the HSI is stopped, HSIRDY flag goes low after 6 HSI oscillator clock cycles.

`_HAL_RCC_HSI_DISABLE`  
`_HAL_RCC_HSI_CALIBRATION`  
`VALUE_ADJUST`

#### Description:

- Macro to adjust the Internal High Speed oscillator (HSI) calibration value.

#### Parameters:

- `_HSICALIBRATIONVALUE_`: specifies the calibration trimming value. (default is `RCC_HSICALIBRATION_DEFAULT`). This parameter must be a number between 0 and 0x1F.

#### Notes:

- The calibration is used to compensate for the variations in voltage and temperature that influence the frequency of the internal HSI RC.

#### *RCC I2C1 Clock Source*

`RCC_I2C1CLKSOURCE_HSI`  
`RCC_I2C1CLKSOURCE_SYSCLK`

#### *RCC I2Cx Clock Config*

`_HAL_RCC_I2C1_CONFIG`

#### Description:

- Macro to configure the I2C1 clock (I2C1CLK).

#### Parameters:

- `_I2C1CLKSOURCE_`: specifies the I2C1 clock source. This parameter can be one of the following values:
  - `RCC_I2C1CLKSOURCE_HSI` HSI selected as I2C1 clock
  - `RCC_I2C1CLKSOURCE_SYSCLK`

System Clock selected as I2C1 clock

**\_\_HAL\_RCC\_GET\_I2C1\_SOURCE    Description:**

- Macro to get the I2C1 clock source.

**Return value:**

- The: clock source can be one of the following values:
  - RCC\_I2C1CLKSOURCE\_HSI HSI selected as I2C1 clock
  - RCC\_I2C1CLKSOURCE\_SYSCLK System Clock selected as I2C1 clock

**Interrupts**

|               |                                      |
|---------------|--------------------------------------|
| RCC_IT_LSIRDY | LSI Ready Interrupt flag             |
| RCC_IT_LSERDY | LSE Ready Interrupt flag             |
| RCC_IT_HSIRDY | HSI Ready Interrupt flag             |
| RCC_IT_HSERDY | HSE Ready Interrupt flag             |
| RCC_IT_PLLRDY | PLL Ready Interrupt flag             |
| RCC_IT_CSS    | Clock Security System Interrupt flag |

**LSE Config**

|                |                                     |
|----------------|-------------------------------------|
| RCC_LSE_OFF    | LSE clock deactivation              |
| RCC_LSE_ON     | LSE clock activation                |
| RCC_LSE_BYPASS | External clock source for LSE clock |

**LSE Configuration**

**\_\_HAL\_RCC\_LSE\_CONFIG    Description:**

- Macro to configure the External Low Speed oscillator (LSE).

**Parameters:**

- \_\_STATE\_\_: specifies the new state of the LSE. This parameter can be one of the following values:
  - RCC\_LSE\_OFF turn OFF the LSE oscillator, LSERDY flag goes low after 6 LSE oscillator clock cycles.
  - RCC\_LSE\_ON turn ON the LSE oscillator.
  - RCC\_LSE\_BYPASS LSE oscillator bypassed with external clock.

**Notes:**

- Transitions LSE Bypass to LSE On and LSE On to LSE Bypass are not supported by this macro. As the LSE is in the Backup domain and write access is denied to this domain after reset, you have to enable write access using HAL\_PWR\_EnableBkUpAccess() function before to configure the LSE (to be done once after reset). After enabling the LSE (RCC\_LSE\_ON or RCC\_LSE\_BYPASS), the application software should

wait on LSERDY flag to be set indicating that LSE clock is stable and can be used to clock the RTC.

#### ***LSI Config***

|             |                        |
|-------------|------------------------|
| RCC_LSI_OFF | LSI clock deactivation |
| RCC_LSI_ON  | LSI clock activation   |

#### ***LSI Configuration***

|                      |               |
|----------------------|---------------|
| __HAL_RCC_LSI_ENABLE | <b>Notes:</b> |
|----------------------|---------------|

- After enabling the LSI, the application software should wait on LSIRDY flag to be set indicating that LSI clock is stable and can be used to clock the IWDG and/or the RTC.

|                       |               |
|-----------------------|---------------|
| __HAL_RCC_LSI_DISABLE | <b>Notes:</b> |
|-----------------------|---------------|

- LSI can not be disabled if the IWDG is running. When the LSI is stopped, LSIRDY flag goes low after 6 LSI oscillator clock cycles.

#### ***MCO Index***

|          |                                                      |
|----------|------------------------------------------------------|
| RCC_MCO1 |                                                      |
| RCC_MCO  | MCO1 to be compliant with other families with 2 MCOs |

#### ***Oscillator Type***

|                         |  |
|-------------------------|--|
| RCC_OSCILLATORTYPE_NONE |  |
| RCC_OSCILLATORTYPE_HSE  |  |
| RCC_OSCILLATORTYPE_HSI  |  |
| RCC_OSCILLATORTYPE_LSE  |  |
| RCC_OSCILLATORTYPE_LSI  |  |

#### ***PLL Clock Source***

|                   |                                                           |
|-------------------|-----------------------------------------------------------|
| RCC_PLLSOURCE_HSI | HSI clock divided by 2 selected as PLL entry clock source |
| RCC_PLLSOURCE_HSE | HSE clock selected as PLL entry clock source              |

#### ***PLL Config***

|              |                       |
|--------------|-----------------------|
| RCC_PLL_NONE | PLL is not configured |
| RCC_PLL_OFF  | PLL deactivation      |
| RCC_PLL_ON   | PLL activation        |

***PLL Configuration***`_HAL_RCC_PLL_ENABLE`**Notes:**

- After enabling the main PLL, the application software should wait on PLLRDY flag to be set indicating that PLL clock is stable and can be used as system clock source. The main PLL is disabled by hardware when entering STOP and STANDBY modes.

`_HAL_RCC_PLL_DISABLE`**Notes:**

- The main PLL can not be disabled if it is used as system clock source

`_HAL_RCC_GET_PLL_OSCSOURCE`**Description:**

- Get oscillator clock selected as PLL input clock.

**Return value:**

- The: clock source used for PLL entry. The returned value can be one of the following:
  - RCC\_PLLSOURCE\_HSI HSI oscillator clock selected as PLL input clock
  - RCC\_PLLSOURCE\_HSE HSE oscillator clock selected as PLL input clock

***RCC PLL HSE Prediv Factor***

`RCC_HSE_PREDIV_DIV1`  
`RCC_HSE_PREDIV_DIV2`  
`RCC_HSE_PREDIV_DIV3`  
`RCC_HSE_PREDIV_DIV4`  
`RCC_HSE_PREDIV_DIV5`  
`RCC_HSE_PREDIV_DIV6`  
`RCC_HSE_PREDIV_DIV7`  
`RCC_HSE_PREDIV_DIV8`  
`RCC_HSE_PREDIV_DIV9`  
`RCC_HSE_PREDIV_DIV10`  
`RCC_HSE_PREDIV_DIV11`  
`RCC_HSE_PREDIV_DIV12`  
`RCC_HSE_PREDIV_DIV13`  
`RCC_HSE_PREDIV_DIV14`  
`RCC_HSE_PREDIV_DIV15`  
`RCC_HSE_PREDIV_DIV16`

**RCC PLL Multiplication Factor**

RCC\_PLL\_MUL2  
RCC\_PLL\_MUL3  
RCC\_PLL\_MUL4  
RCC\_PLL\_MUL5  
RCC\_PLL\_MUL6  
RCC\_PLL\_MUL7  
RCC\_PLL\_MUL8  
RCC\_PLL\_MUL9  
RCC\_PLL\_MUL10  
RCC\_PLL\_MUL11  
RCC\_PLL\_MUL12  
RCC\_PLL\_MUL13  
RCC\_PLL\_MUL14  
RCC\_PLL\_MUL15  
RCC\_PLL\_MUL16

**Register offsets**

RCC\_OFFSET  
RCC\_CR\_OFFSET  
RCC\_CFGR\_OFFSET  
RCC\_CIR\_OFFSET  
RCC\_BDCR\_OFFSET  
RCC\_CSR\_OFFSET

**RCC RTC Clock Configuration**

`__HAL_RCC_RTC_CONFIG`

**Description:**

- Macro to configure the RTC clock (RTCCLK).

**Parameters:**

- `__RTC_CLKSOURCE__`: specifies the RTC clock source. This parameter can be one of the following values:
  - `RCC_RTCCLKSOURCE_NO_CLK` No clock selected as RTC clock
  - `RCC_RTCCLKSOURCE_LSE` LSE selected as RTC clock
  - `RCC_RTCCLKSOURCE_LSI` LSI selected as RTC clock
  - `RCC_RTCCLKSOURCE_HSE_DIV32` HSE clock divided by 32

**Notes:**

- As the RTC clock configuration bits are in the Backup domain and write access is denied to this domain after reset, you have to enable write access using the Power Backup Access macro before to configure the RTC clock source (to be done once after reset). Once the RTC clock is configured it cannot be changed unless the Backup domain is reset using `__HAL_RCC_BACKUPRESET_FORCE()` macro, or by a Power On Reset (POR).
- If the LSE or LSI is used as RTC clock source, the RTC continues to work in STOP and STANDBY modes, and can be used as wakeup source. However, when the LSI clock and HSE clock divided by 32 is used as RTC clock source, the RTC cannot be used in STOP and STANDBY modes. The system must always be configured so as to get a PCLK frequency greater than or equal to the RTCCLK frequency for a proper operation of the RTC.

#### `__HAL_RCC_GET_RTC_SOURCE`

##### **Description:**

- Macro to get the RTC clock source.

##### **Return value:**

- The clock source can be one of the following values:
  - `RCC_RTCCLKSOURCE_NO_CLK` No clock selected as RTC clock
  - `RCC_RTCCLKSOURCE_LSE` LSE selected as RTC clock
  - `RCC_RTCCLKSOURCE_LSI` LSI selected as RTC clock
  - `RCC_RTCCLKSOURCE_HSE_DIV32` HSE clock divided by 32

#### `__HAL_RCC_RTC_ENABLE`

##### **Notes:**

- These macros must be used only after the RTC clock source was selected.

#### `__HAL_RCC_RTC_DISABLE`

##### **Notes:**

- These macros must be used only after the RTC clock source was selected.

#### `__HAL_RCC_BACKUPRESET_FORCE`

##### **Notes:**

- This function resets the RTC peripheral (including the backup registers) and the RTC clock source selection in `RCC_BDCR` register.

---

**\_HAL\_RCC\_BACKUPRESET\_  
RELEASE**

***RTC Clock Source***

|                            |                                                      |
|----------------------------|------------------------------------------------------|
| RCC_RTCCLKSOURCE_NO_CLK    | No clock                                             |
| RCC_RTCCLKSOURCE_LSE       | LSE oscillator clock used as RTC clock               |
| RCC_RTCCLKSOURCE_LSI       | LSI oscillator clock used as RTC clock               |
| RCC_RTCCLKSOURCE_HSE_DIV32 | HSE oscillator clock divided by 32 used as RTC clock |

***System Clock Source***

|                         |                              |
|-------------------------|------------------------------|
| RCC_SYSCLKSOURCE_HSI    | HSI selected as system clock |
| RCC_SYSCLKSOURCE_HSE    | HSE selected as system clock |
| RCC_SYSCLKSOURCE_PLLCLK | PLL selected as system clock |

***System Clock Source Status***

|                                |                          |
|--------------------------------|--------------------------|
| RCC_SYSCLKSOURCE_STATUS_HSI    | HSI used as system clock |
| RCC_SYSCLKSOURCE_STATUS_HSE    | HSE used as system clock |
| RCC_SYSCLKSOURCE_STATUS_PLLCLK | PLL used as system clock |

***System Clock Type***

|                      |                     |
|----------------------|---------------------|
| RCC_CLOCKTYPE_SYSCLK | SYSCLK to configure |
| RCC_CLOCKTYPE_HCLK   | HCLK to configure   |
| RCC_CLOCKTYPE_PCLK1  | PCLK1 to configure  |
| RCC_CLOCKTYPE_PCLK2  | PCLK2 to configure  |

***RCC Timeout***

|                           |  |
|---------------------------|--|
| RCC_DBP_TIMEOUT_VALUE     |  |
| RCC_LSE_TIMEOUT_VALUE     |  |
| CLOCKSWITCH_TIMEOUT_VALUE |  |
| HSE_TIMEOUT_VALUE         |  |
| HSI_TIMEOUT_VALUE         |  |
| LSI_TIMEOUT_VALUE         |  |
| PLL_TIMEOUT_VALUE         |  |

***RCC USART2 Clock Source***

|                            |  |
|----------------------------|--|
| RCC_USART2CLKSOURCE_PCLK1  |  |
| RCC_USART2CLKSOURCE_SYSCLK |  |
| RCC_USART2CLKSOURCE_LSE    |  |
| RCC_USART2CLKSOURCE_HSI    |  |

***RCC USART3 Clock Source***

|                            |  |
|----------------------------|--|
| RCC_USART3CLKSOURCE_PCLK1  |  |
| RCC_USART3CLKSOURCE_SYSCLK |  |

RCC\_USART3CLKSOURCE\_LSE

RCC\_USART3CLKSOURCE\_HSI

### **RCC USARTx Clock Config**

\_\_HAL\_RCC\_USART1\_CONFIG **Description:**

- Macro to configure the USART1 clock (USART1CLK).

#### **Parameters:**

- \_\_USART1CLKSOURCE\_\_: specifies the USART1 clock source. This parameter can be one of the following values:
  - RCC\_USART1CLKSOURCE\_PCLK2 PCLK2 selected as USART1 clock
  - RCC\_USART1CLKSOURCE\_HSI HSI selected as USART1 clock
  - RCC\_USART1CLKSOURCE\_SYSCLK System Clock selected as USART1 clock
  - RCC\_USART1CLKSOURCE\_LSE LSE selected as USART1 clock

\_\_HAL\_RCC\_GET\_USART1\_SOURCE

#### **Description:**

- Macro to get the USART1 clock source.

#### **Return value:**

- The clock source can be one of the following values:
  - RCC\_USART1CLKSOURCE\_PCLK2 PCLK2 selected as USART1 clock
  - RCC\_USART1CLKSOURCE\_HSI HSI selected as USART1 clock
  - RCC\_USART1CLKSOURCE\_SYSCLK System Clock selected as USART1 clock
  - RCC\_USART1CLKSOURCE\_LSE LSE selected as USART1 clock

\_\_HAL\_RCC\_USART2\_CONFIG

#### **Description:**

- Macro to configure the USART2 clock (USART2CLK).

#### **Parameters:**

- \_\_USART2CLKSOURCE\_\_: specifies the USART2 clock source. This parameter can be one of the following values:
  - RCC\_USART2CLKSOURCE\_PCLK1 PCLK1 selected as USART2 clock
  - RCC\_USART2CLKSOURCE\_HSI HSI selected as USART2 clock
  - RCC\_USART2CLKSOURCE\_SYSCLK System Clock selected as USART2 clock
  - RCC\_USART2CLKSOURCE\_LSE LSE selected as USART2 clock

`__HAL_RCC_GET_USART2_SOURCE`

**Description:**

- Macro to get the USART2 clock source.

**Return value:**

- The: clock source can be one of the following values:
  - `RCC_USART2CLKSOURCE_PCLK1` PCLK1 selected as USART2 clock
  - `RCC_USART2CLKSOURCE_HSI` HSI selected as USART2 clock
  - `RCC_USART2CLKSOURCE_SYSCLK` System Clock selected as USART2 clock
  - `RCC_USART2CLKSOURCE_LSE` LSE selected as USART2 clock

`__HAL_RCC_USART3_CONFIG`

**Description:**

- Macro to configure the USART3 clock (USART3CLK).

**Parameters:**

- `__USART3CLKSOURCE__`: specifies the USART3 clock source. This parameter can be one of the following values:
  - `RCC_USART3CLKSOURCE_PCLK1` PCLK1 selected as USART3 clock
  - `RCC_USART3CLKSOURCE_HSI` HSI selected as USART3 clock
  - `RCC_USART3CLKSOURCE_SYSCLK` System Clock selected as USART3 clock
  - `RCC_USART3CLKSOURCE_LSE` LSE selected as USART3 clock

`__HAL_RCC_GET_USART3_SOURCE`

**Description:**

- Macro to get the USART3 clock source.

**Return value:**

- The: clock source can be one of the following values:
  - `RCC_USART3CLKSOURCE_PCLK1` PCLK1 selected as USART3 clock
  - `RCC_USART3CLKSOURCE_HSI` HSI selected as USART3 clock
  - `RCC_USART3CLKSOURCE_SYSCLK` System Clock selected as USART3 clock
  - `RCC_USART3CLKSOURCE_LSE` LSE selected as USART3 clock

# 41 HAL RCC Extension Driver

## 41.1 RCCEEx Firmware driver registers structures

### 41.1.1 RCC\_PерiphCLKInitTypeDef

#### Data Fields

- *uint32\_t PeriphClockSelection*
- *uint32\_t RTCClockSelection*
- *uint32\_t Usart1ClockSelection*
- *uint32\_t Usart2ClockSelection*
- *uint32\_t Usart3ClockSelection*
- *uint32\_t Uart4ClockSelection*
- *uint32\_t Uart5ClockSelection*
- *uint32\_t I2c1ClockSelection*
- *uint32\_t I2c2ClockSelection*
- *uint32\_t Adc12ClockSelection*
- *uint32\_t Adc34ClockSelection*
- *uint32\_t I2sClockSelection*
- *uint32\_t Tim1ClockSelection*
- *uint32\_t Tim8ClockSelection*
- *uint32\_t USBClockSelection*

#### Field Documentation

- *uint32\_t RCC\_PерiphCLKInitTypeDef::PeriphClockSelection*  
The Extended Clock to be configured. This parameter can be a value of [\*RCCEx\\_Periph\\_Clock\\_Selection\*](#)
- *uint32\_t RCC\_PерiphCLKInitTypeDef::RTCClockSelection*  
Specifies RTC Clock Prescalers Selection This parameter can be a value of [\*RCC\\_RTC\\_Clock\\_Source\*](#)
- *uint32\_t RCC\_PерiphCLKInitTypeDef::Usart1ClockSelection*  
USART1 clock source This parameter can be a value of [\*RCCEx\\_USART1\\_Clock\\_Source\*](#)
- *uint32\_t RCC\_PерiphCLKInitTypeDef::Usart2ClockSelection*  
USART2 clock source This parameter can be a value of [\*RCC\\_USART2\\_Clock\\_Source\*](#)
- *uint32\_t RCC\_PерiphCLKInitTypeDef::Usart3ClockSelection*  
USART3 clock source This parameter can be a value of [\*RCC\\_USART3\\_Clock\\_Source\*](#)
- *uint32\_t RCC\_PерiphCLKInitTypeDef::Uart4ClockSelection*  
UART4 clock source This parameter can be a value of [\*RCCEx\\_UART4\\_Clock\\_Source\*](#)
- *uint32\_t RCC\_PерiphCLKInitTypeDef::Uart5ClockSelection*  
UART5 clock source This parameter can be a value of [\*RCCEx\\_UART5\\_Clock\\_Source\*](#)
- *uint32\_t RCC\_PерiphCLKInitTypeDef::I2c1ClockSelection*  
I2C1 clock source This parameter can be a value of [\*RCC\\_I2C1\\_Clock\\_Source\*](#)
- *uint32\_t RCC\_PерiphCLKInitTypeDef::I2c2ClockSelection*  
I2C2 clock source This parameter can be a value of [\*RCCEx\\_I2C2\\_Clock\\_Source\*](#)

- **`uint32_t RCC_PeriphCLKInitTypeDef::Adc12ClockSelection`**  
ADC1 & ADC2 clock source This parameter can be a value of `RCCEEx_ADC12_Clock_Source`
- **`uint32_t RCC_PeriphCLKInitTypeDef::Adc34ClockSelection`**  
ADC3 & ADC4 clock source This parameter can be a value of `RCCEEx_ADC34_Clock_Source`
- **`uint32_t RCC_PeriphCLKInitTypeDef::I2sClockSelection`**  
I2S clock source This parameter can be a value of `RCCEEx_I2S_Clock_Source`
- **`uint32_t RCC_PeriphCLKInitTypeDef::Tim1ClockSelection`**  
TIM1 clock source This parameter can be a value of `RCCEEx_TIM1_Clock_Source`
- **`uint32_t RCC_PeriphCLKInitTypeDef::Tim8ClockSelection`**  
TIM8 clock source This parameter can be a value of `RCCEEx_TIM8_Clock_Source`
- **`uint32_t RCC_PeriphCLKInitTypeDef::USBClockSelection`**  
USB clock source This parameter can be a value of `RCCEEx_USB_Clock_Source`

## 41.2 RCCEEx Firmware driver API description

### 41.2.1 Extended Peripheral Control functions

This subsection provides a set of functions allowing to control the RCC Clocks frequencies.



Important note: Care must be taken when `HAL_RCCEEx_PeriphCLKConfig()` is used to select the RTC clock source; in this case the Backup domain will be reset in order to modify the RTC Clock source, as consequence RTC registers (including the backup registers) are set to their reset values.

This section contains the following APIs:

- `HAL_RCCEEx_PeriphCLKConfig()`
- `HAL_RCCEEx_GetPeriphCLKConfig()`
- `HAL_RCCEEx_GetPeriphCLKFreq()`

### 41.2.2 Detailed description of functions

#### `HAL_RCCEEx_PeriphCLKConfig`

|                      |                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_RCCEEx_PeriphCLKConfig(<br/>(RCC_PeriphCLKInitTypeDef * PeriphClkInit)</code>                                                                                                                                                                                                                                                                           |
| Function description | Initializes the RCC extended peripherals clocks according to the specified parameters in the <code>RCC_PeriphCLKInitTypeDef</code> .                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>PeriphClkInit:</b> pointer to an <code>RCC_PeriphCLKInitTypeDef</code> structure that contains the configuration information for the Extended Peripherals clocks (ADC, CEC, I2C, I2S, SDADC, HRTIM, TIM, USART, RTC and USB).</li> </ul>                                                                                                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                              |
| Notes                | <ul style="list-style-type: none"> <li>• Care must be taken when <code>HAL_RCCEEx_PeriphCLKConfig()</code> is used to select the RTC clock source; in this case the Backup domain will be reset in order to modify the RTC Clock source, as consequence RTC registers (including the backup registers) and <code>RCC_BDCR</code> register are set to their reset values.</li> </ul> |

**HAL\_RCCEEx\_GetPeriphCLKConfig**

|                      |                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void HAL_RCCEEx_GetPeriphCLKConfig (RCC_PeriphCLKInitTypeDef * PeriphClkInit)</code>                                                                                                                                                                                    |
| Function description | Get the RCC_ClkInitStruct according to the internal RCC configuration registers.                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>PeriphClkInit:</b> pointer to an RCC_PeriphCLKInitTypeDef structure that returns the configuration information for the Extended Peripherals clocks (ADC, CEC, I2C, I2S, SDADC, HRTIM, TIM, USART, RTC and USB clocks).</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                               |

**HAL\_RCCEEx\_GetPeriphCLKFreq**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_RCCEEx_GetPeriphCLKFreq (uint32_t PeriphClk)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Returns the peripheral clock frequency.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>PeriphClk:</b> Peripheral clock identifier This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>RCC_PERIPHCLK_RTC</code> RTC peripheral clock</li> <li>- <code>RCC_PERIPHCLK_USART1</code> USART1 peripheral clock</li> <li>- <code>RCC_PERIPHCLK_I2C1</code> I2C1 peripheral clock</li> <li>- <code>RCC_PERIPHCLK_USART2</code> USART2 peripheral clock</li> <li>- <code>RCC_PERIPHCLK_USART3</code> USART3 peripheral clock</li> <li>- <code>RCC_PERIPHCLK_UART4</code> UART4 peripheral clock</li> <li>- <code>RCC_PERIPHCLK_UART5</code> UART5 peripheral clock</li> <li>- <code>RCC_PERIPHCLK_I2C2</code> I2C2 peripheral clock</li> <li>- <code>RCC_PERIPHCLK_I2S</code> I2S peripheral clock</li> <li>- <code>RCC_PERIPHCLK_USB</code> USB peripheral clock</li> <li>- <code>RCC_PERIPHCLK_ADC12</code> ADC12 peripheral clock</li> <li>- <code>RCC_PERIPHCLK_ADC34</code> ADC34 peripheral clock</li> <li>- <code>RCC_PERIPHCLK_TIM1</code> TIM1 peripheral clock</li> <li>- <code>RCC_PERIPHCLK_TIM8</code> TIM8 peripheral clock</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Frequency:</b> in Hz (0: means that no available frequency for the peripheral)</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• Returns 0 if peripheral clock is unknown or 0xDEADDEAD if not applicable.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

## 41.3 RCCEEx Firmware driver defines

### 41.3.1 RCCEEx

***RCC Extended ADC12 Clock Source***

`RCC_ADC12PLLCLK_OFF`  
`RCC_ADC12PLLCLK_DIV1`  
`RCC_ADC12PLLCLK_DIV2`  
`RCC_ADC12PLLCLK_DIV4`  
`RCC_ADC12PLLCLK_DIV6`

---

RCC\_ADC12PLLCLK\_DIV8  
RCC\_ADC12PLLCLK\_DIV10  
RCC\_ADC12PLLCLK\_DIV12  
RCC\_ADC12PLLCLK\_DIV16  
RCC\_ADC12PLLCLK\_DIV32  
RCC\_ADC12PLLCLK\_DIV64  
RCC\_ADC12PLLCLK\_DIV128  
RCC\_ADC12PLLCLK\_DIV256

***RCC Extended ADC34 Clock Source***

RCC\_ADC34PLLCLK\_OFF  
RCC\_ADC34PLLCLK\_DIV1  
RCC\_ADC34PLLCLK\_DIV2  
RCC\_ADC34PLLCLK\_DIV4  
RCC\_ADC34PLLCLK\_DIV6  
RCC\_ADC34PLLCLK\_DIV8  
RCC\_ADC34PLLCLK\_DIV10  
RCC\_ADC34PLLCLK\_DIV12  
RCC\_ADC34PLLCLK\_DIV16  
RCC\_ADC34PLLCLK\_DIV32  
RCC\_ADC34PLLCLK\_DIV64  
RCC\_ADC34PLLCLK\_DIV128  
RCC\_ADC34PLLCLK\_DIV256

***RCC Extended ADCx Clock Config***

\_HAL\_RCC\_ADC12\_CONFIG

**Description:**

- Macro to configure the ADC1 & ADC2 clock (ADC12CLK).

**Parameters:**

- \_ADC12CLKSource\_: specifies the ADC1 & ADC2 clock source. This parameter can be one of the following values:
  - RCC\_ADC12PLLCLK\_OFF ADC1 & ADC2 PLL clock disabled, ADC1 & ADC2 can use AHB clock
  - RCC\_ADC12PLLCLK\_DIV1 PLL clock divided by 1 selected as ADC1 & ADC2 clock
  - RCC\_ADC12PLLCLK\_DIV2 PLL clock divided by 2 selected as ADC1 & ADC2 clock
  - RCC\_ADC12PLLCLK\_DIV4 PLL clock divided by 4 selected as ADC1 & ADC2 clock

- RCC\_ADC12PLLCLK\_DIV6 PLL clock divided by 6 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV8 PLL clock divided by 8 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV10 PLL clock divided by 10 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV12 PLL clock divided by 12 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV16 PLL clock divided by 16 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV32 PLL clock divided by 32 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV64 PLL clock divided by 64 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV128 PLL clock divided by 128 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV256 PLL clock divided by 256 selected as ADC1 & ADC2 clock

[\\_\\_HAL\\_RCC\\_GET\\_ADC12\\_SOURCE](#) **Description:**

- Macro to get the ADC1 & ADC2 clock.

**Return value:**

- The: clock source can be one of the following values:
  - RCC\_ADC12PLLCLK\_OFF ADC1 & ADC2 PLL clock disabled, ADC1 & ADC2 can use AHB clock
  - RCC\_ADC12PLLCLK\_DIV1 PLL clock divided by 1 selected as ADC1 & ADC2 clock
  - RCC\_ADC12PLLCLK\_DIV2 PLL clock divided by 2 selected as ADC1 & ADC2 clock
  - RCC\_ADC12PLLCLK\_DIV4 PLL clock divided by 4 selected as ADC1 & ADC2 clock
  - RCC\_ADC12PLLCLK\_DIV6 PLL clock divided by 6 selected as ADC1 & ADC2 clock
  - RCC\_ADC12PLLCLK\_DIV8 PLL clock divided by 8 selected as ADC1 & ADC2 clock
  - RCC\_ADC12PLLCLK\_DIV10 PLL clock divided by 10 selected as ADC1 &

- ADC2 clock
- RCC\_ADC12PLLCLK\_DIV12 PLL clock divided by 12 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV16 PLL clock divided by 16 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV32 PLL clock divided by 32 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV64 PLL clock divided by 64 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV128 PLL clock divided by 128 selected as ADC1 & ADC2 clock
- RCC\_ADC12PLLCLK\_DIV256 PLL clock divided by 256 selected as ADC1 & ADC2 clock

[\\_\\_HAL\\_RCC\\_ADC34\\_CONFIG](#)**Description:**

- Macro to configure the ADC3 & ADC4 clock (ADC34CLK).

**Parameters:**

- \_\_ADC34CLKSource\_\_: specifies the ADC3 & ADC4 clock source. This parameter can be one of the following values:
  - RCC\_ADC34PLLCLK\_OFF ADC3 & ADC4 PLL clock disabled, ADC3 & ADC4 can use AHB clock
  - RCC\_ADC34PLLCLK\_DIV1 PLL clock divided by 1 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV2 PLL clock divided by 2 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV4 PLL clock divided by 4 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV6 PLL clock divided by 6 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV8 PLL clock divided by 8 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV10 PLL clock divided by 10 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV12 PLL clock divided by 12 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV16 PLL clock divided by 16 selected as ADC3 &

- ADC4 clock
- RCC\_ADC34PLLCLK\_DIV32 PLL clock divided by 32 selected as ADC3 & ADC4 clock
- RCC\_ADC34PLLCLK\_DIV64 PLL clock divided by 64 selected as ADC3 & ADC4 clock
- RCC\_ADC34PLLCLK\_DIV128 PLL clock divided by 128 selected as ADC3 & ADC4 clock
- RCC\_ADC34PLLCLK\_DIV256 PLL clock divided by 256 selected as ADC3 & ADC4 clock

**\_HAL\_RCC\_GET\_ADC34\_SOURCE****Description:**

- Macro to get the ADC3 & ADC4 clock.

**Return value:**

- The: clock source can be one of the following values:
  - RCC\_ADC34PLLCLK\_OFF ADC3 & ADC4 PLL clock disabled, ADC3 & ADC4 can use AHB clock
  - RCC\_ADC34PLLCLK\_DIV1 PLL clock divided by 1 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV2 PLL clock divided by 2 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV4 PLL clock divided by 4 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV6 PLL clock divided by 6 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV8 PLL clock divided by 8 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV10 PLL clock divided by 10 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV12 PLL clock divided by 12 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV16 PLL clock divided by 16 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV32 PLL clock divided by 32 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV64 PLL clock divided by 64 selected as ADC3 & ADC4 clock
  - RCC\_ADC34PLLCLK\_DIV128 PLL

- clock divided by 128 selected as ADC3 & ADC4 clock
- RCC\_ADC34PLLCLK\_DIV256 PLL clock divided by 256 selected as ADC3 & ADC4 clock

**RCC Extended AHB Clock Enable Disable**

```
_HAL_RCC_DMA2_CLK_ENABLE
_HAL_RCC_GPIOE_CLK_ENABLE
_HAL_RCC_ADC12_CLK_ENABLE
_HAL_RCC_ADC1_CLK_ENABLE
_HAL_RCC_ADC2_CLK_ENABLE
_HAL_RCC_DMA2_CLK_DISABLE
_HAL_RCC_GPIOE_CLK_DISABLE
_HAL_RCC_ADC12_CLK_DISABLE
_HAL_RCC_ADC1_CLK_DISABLE
_HAL_RCC_ADC2_CLK_DISABLE
_HAL_RCC_ADC34_CLK_ENABLE
_HAL_RCC_ADC34_CLK_DISABLE
```

**RCC Extended AHB Force Release Reset**

```
_HAL_RCC_GPIOE_FORCE_RESET
_HAL_RCC_ADC12_FORCE_RESET
_HAL_RCC_ADC1_FORCE_RESET
_HAL_RCC_ADC2_FORCE_RESET
_HAL_RCC_GPIOE_RELEASE_RESET
_HAL_RCC_ADC12_RELEASE_RESET
_HAL_RCC_ADC1_RELEASE_RESET
_HAL_RCC_ADC2_RELEASE_RESET
_HAL_RCC_ADC34_FORCE_RESET
_HAL_RCC_ADC34_RELEASE_RESET
```

**RCC Extended AHB Peripheral Clock Enable Disable Status**

```
_HAL_RCC_DMA2_IS_CLK_ENABLED
_HAL_RCC_GPIOE_IS_CLK_ENABLED
_HAL_RCC_ADC12_IS_CLK_ENABLED
_HAL_RCC_DMA2_IS_CLK_DISABLED
_HAL_RCC_GPIOE_IS_CLK_DISABLED
_HAL_RCC_ADC12_IS_CLK_DISABLED
_HAL_RCC_ADC34_IS_CLK_ENABLED
_HAL_RCC_ADC34_IS_CLK_DISABLED
```

**RCC Extended APB1 Clock Enable Disable**

`_HAL_RCC_TIM3_CLK_ENABLE  
_HAL_RCC_TIM4_CLK_ENABLE  
_HAL_RCC_SPI2_CLK_ENABLE  
_HAL_RCC_SPI3_CLK_ENABLE  
_HAL_RCC_UART4_CLK_ENABLE  
_HAL_RCC_UART5_CLK_ENABLE  
_HAL_RCC_I2C2_CLK_ENABLE  
_HAL_RCC_TIM3_CLK_DISABLE  
_HAL_RCC_TIM4_CLK_DISABLE  
_HAL_RCC_SPI2_CLK_DISABLE  
_HAL_RCC_SPI3_CLK_DISABLE  
_HAL_RCC_UART4_CLK_DISABLE  
_HAL_RCC_UART5_CLK_DISABLE  
_HAL_RCC_I2C2_CLK_DISABLE  
_HAL_RCC_TIM7_CLK_ENABLE  
_HAL_RCC_TIM7_CLK_DISABLE  
_HAL_RCC_USB_CLK_ENABLE  
_HAL_RCC_USB_CLK_DISABLE  
_HAL_RCC_CAN1_CLK_ENABLE  
_HAL_RCC_CAN1_CLK_DISABLE`

**RCC Extended APB1 Peripheral Clock Enable Disable Status**

`_HAL_RCC_TIM3_IS_CLK_ENABLED  
_HAL_RCC_TIM4_IS_CLK_ENABLED  
_HAL_RCC_SPI2_IS_CLK_ENABLED  
_HAL_RCC_SPI3_IS_CLK_ENABLED  
_HAL_RCC_UART4_IS_CLK_ENABLED  
_HAL_RCC_UART5_IS_CLK_ENABLED  
_HAL_RCC_I2C2_IS_CLK_ENABLED  
_HAL_RCC_TIM3_IS_CLK_DISABLED  
_HAL_RCC_TIM4_IS_CLK_DISABLED  
_HAL_RCC_SPI2_IS_CLK_DISABLED  
_HAL_RCC_SPI3_IS_CLK_DISABLED  
_HAL_RCC_UART4_IS_CLK_DISABLED  
_HAL_RCC_UART5_IS_CLK_DISABLED  
_HAL_RCC_I2C2_IS_CLK_DISABLED`

\_HAL\_RCC\_TIM7\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_TIM7\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_USB\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_USB\_IS\_CLK\_DISABLED  
\_HAL\_RCC\_CAN1\_IS\_CLK\_ENABLED  
\_HAL\_RCC\_CAN1\_IS\_CLK\_DISABLED

**RCC Extended APB1 Force Release Reset**

\_HAL\_RCC\_TIM3\_FORCE\_RESET  
\_HAL\_RCC\_TIM4\_FORCE\_RESET  
\_HAL\_RCC\_SPI2\_FORCE\_RESET  
\_HAL\_RCC\_SPI3\_FORCE\_RESET  
\_HAL\_RCC\_UART4\_FORCE\_RESET  
\_HAL\_RCC\_UART5\_FORCE\_RESET  
\_HAL\_RCC\_I2C2\_FORCE\_RESET  
\_HAL\_RCC\_TIM3\_RELEASE\_RESET  
\_HAL\_RCC\_TIM4\_RELEASE\_RESET  
\_HAL\_RCC\_SPI2\_RELEASE\_RESET  
\_HAL\_RCC\_SPI3\_RELEASE\_RESET  
\_HAL\_RCC\_UART4\_RELEASE\_RESET  
\_HAL\_RCC\_UART5\_RELEASE\_RESET  
\_HAL\_RCC\_I2C2\_RELEASE\_RESET  
\_HAL\_RCC\_TIM7\_FORCE\_RESET  
\_HAL\_RCC\_TIM7\_RELEASE\_RESET  
\_HAL\_RCC\_USB\_FORCE\_RESET  
\_HAL\_RCC\_USB\_RELEASE\_RESET  
\_HAL\_RCC\_CAN1\_FORCE\_RESET  
\_HAL\_RCC\_CAN1\_RELEASE\_RESET

**RCC Extended APB2 Clock Enable Disable**

\_HAL\_RCC\_SPI1\_CLK\_ENABLE  
\_HAL\_RCC\_SPI1\_CLK\_DISABLE  
\_HAL\_RCC\_TIM8\_CLK\_ENABLE  
\_HAL\_RCC\_TIM8\_CLK\_DISABLE  
\_HAL\_RCC\_TIM1\_CLK\_ENABLE  
\_HAL\_RCC\_TIM1\_CLK\_DISABLE

**RCC Extended APB2 Peripheral Clock Enable Disable Status**

\_HAL\_RCC\_SPI1\_IS\_CLK\_ENABLED

---

`_HAL_RCC_SPI1_IS_CLK_DISABLED`  
`_HAL_RCC_TIM8_IS_CLK_ENABLED`  
`_HAL_RCC_TIM8_IS_CLK_DISABLED`  
`_HAL_RCC_TIM1_IS_CLK_ENABLED`  
`_HAL_RCC_TIM1_IS_CLK_DISABLED`

**RCC Extended APB2 Force Release Reset**

`_HAL_RCC_SPI1_FORCE_RESET`  
`_HAL_RCC_SPI1_RELEASE_RESET`  
`_HAL_RCC_TIM8_FORCE_RESET`  
`_HAL_RCC_TIM8_RELEASE_RESET`  
`_HAL_RCC_TIM1_FORCE_RESET`  
`_HAL_RCC_TIM1_RELEASE_RESET`

**RCC Extended HSE Configuration**

|                                                                                                                                           |                     |
|-------------------------------------------------------------------------------------------------------------------------------------------|---------------------|
| <code>_HAL_RCC_HSE_PREDIV_CONFIG</code>                                                                                                   | <b>Description:</b> |
| <ul style="list-style-type: none"> <li>Macro to configure the External High Speed oscillator (HSE) Predivision factor for PLL.</li> </ul> |                     |

**Parameters:**

- `_HSE_PREDIV_VALUE_`: specifies the division value applied to HSE. This parameter must be a number between `RCC_HSE_PREDIV_DIV1` and `RCC_HSE_PREDIV_DIV16`.

**Notes:**

- Predivision factor can not be changed if PLL is used as system clock In this case, you have to select another source of the system clock, disable the PLL and then change the HSE predivision factor.

`_HAL_RCC_HSE_GET_PREDIV`

**RCC Extended I2C2 Clock Source**

`RCC_I2C2CLKSOURCE_HSI`  
`RCC_I2C2CLKSOURCE_SYSCLK`

**RCC Extended I2Cx Clock Config**

|                                                                                                |                     |
|------------------------------------------------------------------------------------------------|---------------------|
| <code>_HAL_RCC_I2C2_CONFIG</code>                                                              | <b>Description:</b> |
| <ul style="list-style-type: none"> <li>Macro to configure the I2C2 clock (I2C2CLK).</li> </ul> |                     |

**Parameters:**

- `_I2C2CLKSource_`: specifies the I2C2 clock source. This parameter can be one of the following values:
  - `RCC_I2C2CLKSOURCE_HSI` HSI selected as I2C2 clock

- RCC\_I2C2CLKSOURCE\_SYSCLK  
System Clock selected as I2C2 clock

**\_HAL\_RCC\_GET\_I2C2\_SOURCE** **Description:**

- Macro to get the I2C2 clock source.

**Return value:**

- The: clock source can be one of the following values:
  - RCC\_I2C2CLKSOURCE\_HSI HSI selected as I2C2 clock
  - RCC\_I2C2CLKSOURCE\_SYSCLK System Clock selected as I2C2 clock

***RCC Extended I2Sx Clock Config***

**\_HAL\_RCC\_I2S\_CONFIG**

**Description:**

- Macro to configure the I2S clock source (I2SCLK).

**Parameters:**

- I2SCLKSource: specifies the I2S clock source. This parameter can be one of the following values:
  - RCC\_I2SCLKSOURCE\_SYSCLK SYSCLK clock used as I2S clock source
  - RCC\_I2SCLKSOURCE\_EXT External clock mapped on the I2S\_CKIN pin used as I2S clock source

**Notes:**

- This function must be called before enabling the I2S APB clock.

**\_HAL\_RCC\_GET\_I2S\_SOURCE**

**Description:**

- Macro to get the I2S clock source (I2SCLK).

**Return value:**

- The: clock source can be one of the following values:
  - RCC\_I2SCLKSOURCE\_SYSCLK SYSCLK clock used as I2S clock source
  - RCC\_I2SCLKSOURCE\_EXT External clock mapped on the I2S\_CKIN pin used as I2S clock source

***RCC Extended I2S Clock Source***

RCC\_I2SCLKSOURCE\_SYSCLK

RCC\_I2SCLKSOURCE\_EXT

***RCC LSE Drive Configuration***

RCC\_LSEDRIVE\_LOW                   Xtal mode lower driving capability

RCC\_LSEDRIVE\_MEDIUMLOW           Xtal mode medium low driving capability

---

|                                      |                                          |
|--------------------------------------|------------------------------------------|
| <code>RCC_LSEDRIVE_MEDIUMHIGH</code> | Xtal mode medium high driving capability |
| <code>RCC_LSEDRIVE_HIGH</code>       | Xtal mode higher driving capability      |

**LSE Drive Configuration****\_HAL\_RCC\_LSEDRIVE\_CONFIG** **Description:**

- Macro to configure the External Low Speed oscillator (LSE) drive capability.

**Parameters:**

- `_RCC_LSEDRIVE_`: specifies the new state of the LSE drive capability. This parameter can be one of the following values:
  - `RCC_LSEDRIVE_LOW` LSE oscillator low drive capability.
  - `RCC_LSEDRIVE_MEDIUMLOW` LSE oscillator medium low drive capability.
  - `RCC_LSEDRIVE_MEDIUMHIGH` LSE oscillator medium high drive capability.
  - `RCC_LSEDRIVE_HIGH` LSE oscillator high drive capability.

**Return value:**

- None

***RCC Extended MCOx Clock Config*****\_HAL\_RCC\_MCO1\_CONFIG** **Description:**

- Macro to configure the MCO clock.

**Parameters:**

- `_MCOCLKSOURCE_`: specifies the MCO clock source. This parameter can be one of the following values:
  - `RCC_MCO1SOURCE_NOCLOCK` No clock selected as MCO clock
  - `RCC_MCO1SOURCE_SYSCLK` System Clock selected as MCO clock
  - `RCC_MCO1SOURCE_HSI` HSI selected as MCO clock
  - `RCC_MCO1SOURCE_HSE` HSE selected as MCO clock
  - `RCC_MCO1SOURCE_LSI` LSI selected as MCO clock
  - `RCC_MCO1SOURCE_LSE` LSE selected as MCO clock
  - `RCC_MCO1SOURCE_PLLCLK_DIV2` PLLCLK Divided by 2 selected as MCO clock
- `_MCODIV_`: specifies the MCO clock prescaler. This parameter can be one of the following values:
  - `RCC_MCODIV_1` No division applied on MCO clock source

**RCC Extended MCOx Clock Prescaler**`RCC_MCODIV_1`**RCC Extended MCO Clock Source**`RCC_MCO1SOURCE_NOCLOCK``RCC_MCO1SOURCE_LSI``RCC_MCO1SOURCE_LSE``RCC_MCO1SOURCE_SYSCLK``RCC_MCO1SOURCE_HSI``RCC_MCO1SOURCE_HSE``RCC_MCO1SOURCE_PLLCLK_DIV2`**RCC Extended Periph Clock Selection**`RCC_PERIPHCLK_USART1``RCC_PERIPHCLK_USART2``RCC_PERIPHCLK_USART3``RCC_PERIPHCLK_UART4``RCC_PERIPHCLK_UART5``RCC_PERIPHCLK_I2C1``RCC_PERIPHCLK_I2C2``RCC_PERIPHCLK_ADC12``RCC_PERIPHCLK_ADC34``RCC_PERIPHCLK_I2S``RCC_PERIPHCLK_TIM1``RCC_PERIPHCLK_TIM8``RCC_PERIPHCLK_RTC``RCC_PERIPHCLK_USB`**RCC Extended PLL Configuration****\_HAL\_RCC\_PLL\_CONFIG Description:**

- Macro to configure the PLL clock source and multiplication factor.

**Parameters:**

- \_RCC\_PLLSource: specifies the PLL entry clock source. This parameter can be one of the following values:
  - `RCC_PLLSOURCE_HSI` HSI oscillator clock selected as PLL clock entry
  - `RCC_PLLSOURCE_HSE` HSE oscillator clock selected as PLL clock entry
- \_PLLMUL: specifies the multiplication factor for PLL VCO input clock. This parameter must be a number between `RCC_PLL_MUL2` and `RCC_PLL_MUL16`.

**Notes:**

- This macro must be used only when the PLL is disabled.

**RCC Extended TIM1 Clock Source**`RCC_TIM1CLK_HCLK``RCC_TIM1CLK_PLLCLK`**RCC Extended TIM8 Clock Source**`RCC_TIM8CLK_HCLK``RCC_TIM8CLK_PLLCLK`**RCC Extended TIMx Clock Config**`_HAL_RCC_TIM1_CONFIG`**Description:**

- Macro to configure the TIM1 clock (TIM1CLK).

**Parameters:**

- `_TIM1CLKSource_`: specifies the TIM1 clock source. This parameter can be one of the following values:
  - `RCC_TIM1CLK_HCLK` HCLK selected as TIM1 clock
  - `RCC_TIM1CLK_PLLCLK` PLL Clock selected as TIM1 clock

`_HAL_RCC_GET_TIM1_SOURCE`**Description:**

- Macro to get the TIM1 clock (TIM1CLK).

**Return value:**

- The: clock source can be one of the following values:
  - `RCC_TIM1CLK_HCLK` HCLK selected as TIM1 clock
  - `RCC_TIM1CLK_PLLCLK` PLL Clock selected as TIM1 clock

`_HAL_RCC_TIM8_CONFIG`**Description:**

- Macro to configure the TIM8 clock (TIM8CLK).

**Parameters:**

- `_TIM8CLKSource_`: specifies the TIM8 clock source. This parameter can be one of the following values:
  - `RCC_TIM8CLK_HCLK` HCLK selected as TIM8 clock
  - `RCC_TIM8CLK_PLLCLK` PLL Clock selected as TIM8 clock

`_HAL_RCC_GET_TIM8_SOURCE`**Description:**

- Macro to get the TIM8 clock (TIM8CLK).

**Return value:**

- The clock source can be one of the following values:
  - RCC\_TIM8CLK\_HCLK HCLK selected as TIM8 clock
  - RCC\_TIM8CLK\_PLLCLK PLL Clock selected as TIM8 clock

#### **RCC Extended UART4 Clock Source**

RCC\_UART4CLKSOURCE\_PCLK1  
 RCC\_UART4CLKSOURCE\_SYSCLK  
 RCC\_UART4CLKSOURCE\_LSE  
 RCC\_UART4CLKSOURCE\_HSI

#### **RCC Extended UART5 Clock Source**

RCC\_UART5CLKSOURCE\_PCLK1  
 RCC\_UART5CLKSOURCE\_SYSCLK  
 RCC\_UART5CLKSOURCE\_LSE  
 RCC\_UART5CLKSOURCE\_HSI

#### **RCC Extended UARTx Clock Config**

`__HAL_RCC_UART4_CONFIG`

##### **Description:**

- Macro to configure the UART4 clock (UART4CLK).

##### **Parameters:**

- `__UART4CLKSource__`: specifies the UART4 clock source. This parameter can be one of the following values:
  - RCC\_UART4CLKSOURCE\_PCLK1 PCLK1 selected as UART4 clock
  - RCC\_UART4CLKSOURCE\_HSI HSI selected as UART4 clock
  - RCC\_UART4CLKSOURCE\_SYSCLK System Clock selected as UART4 clock
  - RCC\_UART4CLKSOURCE\_LSE LSE selected as UART4 clock

`__HAL_RCC_GET_UART4_SOURCE`

##### **Description:**

- Macro to get the UART4 clock source.

##### **Return value:**

- The clock source can be one of the following values:
  - RCC\_UART4CLKSOURCE\_PCLK1 PCLK1 selected as UART4 clock
  - RCC\_UART4CLKSOURCE\_HSI HSI selected as UART4 clock
  - RCC\_UART4CLKSOURCE\_SYSCLK System Clock selected as UART4 clock
  - RCC\_UART4CLKSOURCE\_LSE LSE selected as UART4 clock

selected as UART4 clock

`_HAL_RCC_UART5_CONFIG`

**Description:**

- Macro to configure the UART5 clock (UART5CLK).

**Parameters:**

- `_UART5CLKSource`: specifies the UART5 clock source. This parameter can be one of the following values:
  - `RCC_UART5CLKSOURCE_PCLK1` PCLK1 selected as UART5 clock
  - `RCC_UART5CLKSOURCE_HSI` HSI selected as UART5 clock
  - `RCC_UART5CLKSOURCE_SYSCLK` System Clock selected as UART5 clock
  - `RCC_UART5CLKSOURCE_LSE` LSE selected as UART5 clock

`_HAL_RCC_GET_UART5_SOURCE`

**Description:**

- Macro to get the UART5 clock source.

**Return value:**

- The clock source can be one of the following values:
  - `RCC_UART5CLKSOURCE_PCLK1` PCLK1 selected as UART5 clock
  - `RCC_UART5CLKSOURCE_HSI` HSI selected as UART5 clock
  - `RCC_UART5CLKSOURCE_SYSCLK` System Clock selected as UART5 clock
  - `RCC_UART5CLKSOURCE_LSE` LSE selected as UART5 clock

***RCC Extended USART1 Clock Source***

`RCC_USART1CLKSOURCE_PCLK2`

`RCC_USART1CLKSOURCE_SYSCLK`

`RCC_USART1CLKSOURCE_LSE`

`RCC_USART1CLKSOURCE_HSI`

**RCC Extended USBx Clock Config**

`__HAL_RCC_USB_CONFIG` **Description:**

- Macro to configure the USB clock (USBCLK).

**Parameters:**

- `__USBCLKSource__`: specifies the USB clock source. This parameter can be one of the following values:
  - `RCC_USBCLKSOURCE_PLL` PLL Clock divided by 1 selected as USB clock
  - `RCC_USBCLKSOURCE_PLL_DIV1_5` PLL Clock divided by 1.5 selected as USB clock

`__HAL_RCC_GET_USB_SOURCE`

**Description:**

- Macro to get the USB clock source.

**Return value:**

- The: clock source can be one of the following values:
  - `RCC_USBCLKSOURCE_PLL` PLL Clock divided by 1 selected as USB clock
  - `RCC_USBCLKSOURCE_PLL_DIV1_5` PLL Clock divided by 1.5 selected as USB clock

**RCC Extended USB Clock Source**

`RCC_USBCLKSOURCE_PLL`

`RCC_USBCLKSOURCE_PLL_DIV1_5`

## 42 HAL RTC Generic Driver

### 42.1 RTC Firmware driver registers structures

#### 42.1.1 RTC\_InitTypeDef

##### Data Fields

- *uint32\_t HourFormat*
- *uint32\_t AsynchPrediv*
- *uint32\_t SynchPrediv*
- *uint32\_t OutPut*
- *uint32\_t OutPutPolarity*
- *uint32\_t OutPutType*

##### Field Documentation

- *uint32\_t RTC\_InitTypeDef::HourFormat*  
Specifies the RTC Hour Format. This parameter can be a value of [\*RTC\\_Hour\\_Formats\*](#)
- *uint32\_t RTC\_InitTypeDef::AsynchPrediv*  
Specifies the RTC Asynchronous Predivider value. This parameter must be a number between Min\_Data = 0x00 and Max\_Data = 0x7F
- *uint32\_t RTC\_InitTypeDef::SynchPrediv*  
Specifies the RTC Synchronous Predivider value. This parameter must be a number between Min\_Data = 0x00 and Max\_Data = 0x7FFF
- *uint32\_t RTC\_InitTypeDef::OutPut*  
Specifies which signal will be routed to the RTC output. This parameter can be a value of [\*RTCEx\\_Output\\_selection\\_Definitions\*](#)
- *uint32\_t RTC\_InitTypeDef::OutPutPolarity*  
Specifies the polarity of the output signal. This parameter can be a value of [\*RTC\\_Output\\_Polarity\\_Definitions\*](#)
- *uint32\_t RTC\_InitTypeDef::OutPutType*  
Specifies the RTC Output Pin mode. This parameter can be a value of [\*RTC\\_Output\\_Type\\_ALARM\\_OUT\*](#)

#### 42.1.2 RTC\_TimeTypeDef

##### Data Fields

- *uint8\_t Hours*
- *uint8\_t Minutes*
- *uint8\_t Seconds*
- *uint8\_t TimeFormat*
- *uint32\_t SubSeconds*
- *uint32\_t SecondFraction*
- *uint32\_t DayLightSaving*
- *uint32\_t StoreOperation*

##### Field Documentation

- *uint8\_t RTC\_TimeTypeDef::Hours*  
Specifies the RTC Time Hour. This parameter must be a number between Min\_Data = 0 and Max\_Data = 12 if the RTC\_HourFormat\_12 is selected. This parameter must be

- a number between Min\_Data = 0 and Max\_Data = 23 if the RTC\_HourFormat\_24 is selected
- ***uint8\_t RTC\_TimeTypeDef::Minutes***  
Specifies the RTC Time Minutes. This parameter must be a number between Min\_Data = 0 and Max\_Data = 59
  - ***uint8\_t RTC\_TimeTypeDef::Seconds***  
Specifies the RTC Time Seconds. This parameter must be a number between Min\_Data = 0 and Max\_Data = 59
  - ***uint8\_t RTC\_TimeTypeDef::TimeFormat***  
Specifies the RTC AM/PM Time. This parameter can be a value of [RTC\\_AM\\_PM\\_Definitions](#)
  - ***uint32\_t RTC\_TimeTypeDef::SubSeconds***  
Specifies the RTC\_SSR RTC Sub Second register content. This parameter corresponds to a time unit range between [0-1] Second with [1 Sec / SecondFraction +1] granularity
  - ***uint32\_t RTC\_TimeTypeDef::SecondFraction***  
Specifies the range or granularity of Sub Second register content corresponding to Synchronous pre-scaler factor value (PREDIV\_S) This parameter corresponds to a time unit range between [0-1] Second with [1 Sec / SecondFraction +1] granularity. This field will be used only by HAL\_RTC\_GetTime function
  - ***uint32\_t RTC\_TimeTypeDef::DayLightSaving***  
Specifies RTC\_DayLightSaveOperation: the value of hour adjustment. This parameter can be a value of [RTC\\_DayLightSaving\\_Definitions](#)
  - ***uint32\_t RTC\_TimeTypeDef::StoreOperation***  
Specifies RTC\_StoreOperation value to be written in the BCK bit in CR register to store the operation. This parameter can be a value of [RTC\\_StoreOperation\\_Definitions](#)

#### 42.1.3 RTC\_DateTypeDef

##### Data Fields

- ***uint8\_t WeekDay***
- ***uint8\_t Month***
- ***uint8\_t Date***
- ***uint8\_t Year***

##### Field Documentation

- ***uint8\_t RTC\_DateTypeDef::WeekDay***  
Specifies the RTC Date WeekDay. This parameter can be a value of [RTC\\_WeekDay\\_Definitions](#)
- ***uint8\_t RTC\_DateTypeDef::Month***  
Specifies the RTC Date Month (in BCD format). This parameter can be a value of [RTC\\_Month\\_Date\\_Definitions](#)
- ***uint8\_t RTC\_DateTypeDef::Date***  
Specifies the RTC Date. This parameter must be a number between Min\_Data = 1 and Max\_Data = 31
- ***uint8\_t RTC\_DateTypeDef::Year***  
Specifies the RTC Date Year. This parameter must be a number between Min\_Data = 0 and Max\_Data = 99

#### 42.1.4 RTC\_AlarmTypeDef

##### Data Fields

- ***RTC\_TimeTypeDef AlarmTime***

- *uint32\_t AlarmMask*
- *uint32\_t AlarmSubSecondMask*
- *uint32\_t AlarmDateWeekDaySel*
- *uint8\_t AlarmDateWeekDay*
- *uint32\_t Alarm*

#### Field Documentation

- ***RTC\_TimeTypeDef RTC\_AlarmTypeDef::AlarmTime***  
Specifies the RTC Alarm Time members
- ***uint32\_t RTC\_AlarmTypeDef::AlarmMask***  
Specifies the RTC Alarm Masks. This parameter can be a value of [\*RTC\\_AlarmMask\\_Definitions\*](#)
- ***uint32\_t RTC\_AlarmTypeDef::AlarmSubSecondMask***  
Specifies the RTC Alarm SubSeconds Masks. This parameter can be a value of [\*RTC\\_Alarm\\_Sub\\_Seconds\\_Masks\\_Definitions\*](#)
- ***uint32\_t RTC\_AlarmTypeDef::AlarmDateWeekDaySel***  
Specifies the RTC Alarm is on Date or WeekDay. This parameter can be a value of [\*RTC\\_AlarmDateWeekDay\\_Definitions\*](#)
- ***uint8\_t RTC\_AlarmTypeDef::AlarmDateWeekDay***  
Specifies the RTC Alarm Date/WeekDay. If the Alarm Date is selected, this parameter must be set to a value in the 1-31 range. If the Alarm WeekDay is selected, this parameter can be a value of [\*RTC\\_WeekDay\\_Definitions\*](#)
- ***uint32\_t RTC\_AlarmTypeDef::Alarm***  
Specifies the alarm . This parameter can be a value of [\*RTC\\_Alarms\\_Definitions\*](#)

### 42.1.5 RTC\_HandleTypeDef

#### Data Fields

- *RTC\_TypeDef \* Instance*
- *RTC\_InitTypeDef Init*
- *HAL\_LockTypeDef Lock*
- *\_\_IO HAL\_RTCStateTypeDef State*

#### Field Documentation

- ***RTC\_TypeDef\* RTC\_HandleTypeDef::Instance***  
Register base address
- ***RTC\_InitTypeDef RTC\_HandleTypeDef::Init***  
RTC required parameters
- ***HAL\_LockTypeDef RTC\_HandleTypeDef::Lock***  
RTC locking object
- ***\_\_IO HAL\_RTCStateTypeDef RTC\_HandleTypeDef::State***  
Time communication state

## 42.2 RTC Firmware driver API description

### 42.2.1 RTC Operating Condition

The real-time clock (RTC) and the RTC backup registers can be powered from the VBAT voltage when the main VDD supply is powered off. To retain the content of the RTC backup registers and supply the RTC when VDD is turned off, VBAT pin can be connected to an optional standby voltage supplied by a battery or by another source.

To allow the RTC to operate even when the main digital supply (VDD) is turned off, the VBAT pin powers the following blocks:

1. The RTC
2. The LSE oscillator
3. PC13 to PC15 I/Os (when available)

When the backup domain is supplied by VDD (analog switch connected to VDD), the following functions are available:

1. PC14 and PC15 can be used as either GPIO or LSE pins
2. PC13 can be used as a GPIO or as the RTC\_OUT pin

When the backup domain is supplied by VBAT (analog switch connected to VBAT because VDD is not present), the following functions are available:

1. PC14 and PC15 can be used as LSE pins only
2. PC13 can be used as the RTC\_OUT pin

#### 42.2.2 Backup Domain Reset

The backup domain reset sets all RTC registers and the RCC\_BDCR register to their reset values. A backup domain reset is generated when one of the following events occurs:

1. Software reset, triggered by setting the BDRST bit in the RCC Backup domain control register (RCC\_BDCR).
2. VDD or VBAT power on, if both supplies have previously been powered off.

#### 42.2.3 Backup Domain Access

After reset, the backup domain (RTC registers, RTC backup data registers and backup SRAM) is protected against possible unwanted write accesses.

To enable access to the RTC Domain and RTC registers, proceed as follows:

1. Enable the Power Controller (PWR) APB1 interface clock using the \_\_HAL\_RCC\_PWR\_CLK\_ENABLE() function.
2. Enable access to RTC domain using the HAL\_PWR\_EnableBkUpAccess() function.
3. Select the RTC clock source using the \_\_HAL\_RCC\_RTC\_CONFIG() function.
4. Enable RTC Clock using the \_\_HAL\_RCC\_RTC\_ENABLE() function.

#### 42.2.4 How to use RTC Driver

- Enable the RTC domain access (see description in the section above).
- Configure the RTC Prescaler (Asynchronous and Synchronous) and RTC hour format using the HAL\_RTC\_Init() function.

##### Time and Date configuration

- To configure the RTC Calendar (Time and Date) use the HAL\_RTC\_SetTime() and HAL\_RTC\_SetDate() functions.
- To read the RTC Calendar, use the HAL\_RTC\_GetTime() and HAL\_RTC\_GetDate() functions.

##### Alarm configuration

- To configure the RTC Alarm use the HAL\_RTC\_SetAlarm() function. You can also configure the RTC Alarm with interrupt mode using the HAL\_RTC\_SetAlarm\_IT() function.
- To read the RTC Alarm, use the HAL\_RTC\_GetAlarm() function.

### RTC Wakeup configuration

- To configure the RTC Wakeup Clock source and Counter use the HAL\_RTC\_SetWakeUpTimer() function. You can also configure the RTC Wakeup timer with interrupt mode using the HAL\_RTC\_SetWakeUpTimer\_IT() function.
- To read the RTC WakeUp Counter register, use the HAL\_RTC\_GetWakeUpTimer() function.

### TimeStamp configuration

- Configure the RTC\_AF trigger and enables the RTC TimeStamp using the HAL\_RTC\_SetTimeStamp() function. You can also configure the RTC TimeStamp with interrupt mode using the HAL\_RTC\_SetTimeStamp\_IT() function.
- To read the RTC TimeStamp Time and Date register, use the HAL\_RTC\_GetTimeStamp() function.

### Tamper configuration

- Enable the RTC Tamper and Configure the Tamper filter count, trigger Edge or Level according to the Tamper filter (if equal to 0 Edge else Level) value, sampling frequency, precharge or discharge and Pull-UP using the HAL\_RTC\_SetTamper() function. You can configure RTC Tamper with interrupt mode using HAL\_RTC\_SetTamper\_IT() function.

### Backup Data Registers configuration

- To write to the RTC Backup Data registers, use the HAL\_RTC\_BKUPWrite() function.
- To read the RTC Backup Data registers, use the HAL\_RTC\_BKUPRead() function.

## 42.2.5 RTC and low power modes

The MCU can be woken up from a low power mode by an RTC alternate function.

The RTC alternate functions are the RTC alarms (Alarm A and Alarm B), RTC wakeup, RTC tamper event detection and RTC time stamp event detection. These RTC alternate functions can wake up the system from the Stop and Standby low power modes.

The system can also wake up from low power modes without depending on an external interrupt (Auto-wakeup mode), by using the RTC alarm or the RTC wakeup events.

The RTC provides a programmable time base for waking up from the Stop or Standby mode at regular intervals. Wakeup from STOP and Standby modes is possible only when the RTC clock source is LSE or LSI.

## 42.2.6 Initialization and de-initialization functions

This section provides functions allowing to initialize and configure the RTC Prescaler (Synchronous and Asynchronous), RTC Hour format, disable RTC registers Write protection, enter and exit the RTC initialization mode, RTC registers synchronization check and reference clock detection enable.

1. The RTC Prescaler is programmed to generate the RTC 1Hz time base. It is split into 2 programmable prescalers to minimize power consumption.
  - A 7-bit asynchronous prescaler and a 15-bit synchronous prescaler.
  - When both prescalers are used, it is recommended to configure the asynchronous prescaler to a high value to minimize power consumption.
2. All RTC registers are Write protected. Writing to the RTC registers is enabled by writing a key into the Write Protection register, RTC\_WPR.

3. To configure the RTC Calendar, user application should enter initialization mode. In this mode, the calendar counter is stopped and its value can be updated. When the initialization sequence is complete, the calendar restarts counting after 4 RTCCLK cycles.
4. To read the calendar through the shadow registers after Calendar initialization, calendar update or after wakeup from low power modes the software must first clear the RSF flag. The software must then wait until it is set again before reading the calendar, which means that the calendar registers have been correctly copied into the RTC\_TR and RTC\_DR shadow registers. The HAL\_RTC\_WaitForSynchro() function implements the above software sequence (RSF clear and RSF check).

This section contains the following APIs:

- [\*HAL\\_RTC\\_Init\(\)\*](#)
- [\*HAL\\_RTC\\_DeInit\(\)\*](#)
- [\*HAL\\_RTC\\_MspInit\(\)\*](#)
- [\*HAL\\_RTC\\_MspDeInit\(\)\*](#)

#### **42.2.7 RTC Time and Date functions**

This section provides functions allowing to configure Time and Date features

This section contains the following APIs:

- [\*HAL\\_RTC\\_SetTime\(\)\*](#)
- [\*HAL\\_RTC\\_GetTime\(\)\*](#)
- [\*HAL\\_RTC\\_SetDate\(\)\*](#)
- [\*HAL\\_RTC\\_GetDate\(\)\*](#)

#### **42.2.8 RTC Alarm functions**

This section provides functions allowing to configure Alarm feature

This section contains the following APIs:

- [\*HAL\\_RTC\\_SetAlarm\(\)\*](#)
- [\*HAL\\_RTC\\_SetAlarm\\_IT\(\)\*](#)
- [\*HAL\\_RTC\\_DeactivateAlarm\(\)\*](#)
- [\*HAL\\_RTC\\_GetAlarm\(\)\*](#)
- [\*HAL\\_RTC\\_AlarmIRQHandler\(\)\*](#)
- [\*HAL\\_RTC\\_AlarmAEventCallback\(\)\*](#)
- [\*HAL\\_RTC\\_PollForAlarmAEvent\(\)\*](#)

#### **42.2.9 Detailed description of functions**

##### **HAL\_RTC\_Init**

|                      |                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_Init (RTC_HandleTypeDef * hrtc)</b>                                                                |
| Function description | Initialize the RTC according to the specified parameters in the RTC_InitTypeDef structure and initialize the associated handle. |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                          |

**HAL\_RTC\_DeInit**

|                      |                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_DeInit (RTC_HandleTypeDef * hrtc)</b>                                             |
| Function description | DeInitialize the RTC peripheral.                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                         |
| Notes                | <ul style="list-style-type: none"> <li>• This function doesn't reset the RTC Backup Data registers.</li> </ul> |

**HAL\_RTC\_MspInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTC_MspInit (RTC_HandleTypeDef * hrtc)</b>                      |
| Function description | Initialize the RTC MSP.                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_RTC\_MspDeInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTC_MspDeInit (RTC_HandleTypeDef * hrtc)</b>                    |
| Function description | DeInitialize the RTC MSP.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_RTC\_SetTime**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_SetTime (RTC_HandleTypeDef * hrtc, RTC_TimeTypeDef * sTime, uint32_t Format)</b>                                                                                                                                                                                                                                                                                  |
| Function description | Set RTC current time.                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>sTime:</b> Pointer to Time structure</li> <li>• <b>Format:</b> Specifies the format of the entered parameters. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– RTC_FORMAT_BIN: Binary data format</li> <li>– RTC_FORMAT_BCD: BCD data format</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                         |

**HAL\_RTC\_GetTime**

|                      |                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_GetTime (RTC_HandleTypeDef * hrtc, RTC_TimeTypeDef * sTime, uint32_t Format)</b>                                                                                                                                                                                                                                       |
| Function description | Get RTC current time.                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>sTime:</b> Pointer to Time structure</li> <li>• <b>Format:</b> Specifies the format of the entered parameters. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– RTC_FORMAT_BIN: Binary data format</li> </ul> </li> </ul> |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>- RTC_FORMAT_BCD: BCD data format</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes         | <ul style="list-style-type: none"> <li>• You can use SubSeconds and SecondFraction (sTime structure fields returned) to convert SubSeconds value in second fraction ratio with time unit following generic formula:<br/>Second fraction ratio * time_unit= [(SecondFraction-SubSeconds)/(SecondFraction+1)] * time_unit This conversion can be performed only if no shift operation is pending (ie. SHFP=0) when PREDIV_S &gt;= SS</li> <li>• Call HAL_RTC_GetDate() after HAL_RTC_GetTime() to unlock the values in the higher-order calendar shadow registers.</li> </ul> |

### HAL\_RTC\_SetDate

|                      |                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_SetDate (RTC_HandleTypeDef * hrtc, RTC_DateTypeDef * sDate, uint32_t Format)</b>                                                                                                                                                                                                                                                                                  |
| Function description | Set RTC current date.                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>sDate:</b> Pointer to date structure</li> <li>• <b>Format:</b> specifies the format of the entered parameters. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- RTC_FORMAT_BIN: Binary data format</li> <li>- RTC_FORMAT_BCD: BCD data format</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                         |

### HAL\_RTC\_GetDate

|                      |                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_GetDate (RTC_HandleTypeDef * hrtc, RTC_DateTypeDef * sDate, uint32_t Format)</b>                                                                                                                                                                                                                                                                                    |
| Function description | Get RTC current date.                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>sDate:</b> Pointer to Date structure</li> <li>• <b>Format:</b> Specifies the format of the entered parameters. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- RTC_FORMAT_BIN : Binary data format</li> <li>- RTC_FORMAT_BCD : BCD data format</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                           |

### HAL\_RTC\_SetAlarm

|                      |                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_SetAlarm (RTC_HandleTypeDef * hrtc, RTC_AlarmTypeDef * sAlarm, uint32_t Format)</b>                                                                                                                                              |
| Function description | Set the specified RTC Alarm.                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>sAlarm:</b> Pointer to Alarm structure</li> <li>• <b>Format:</b> Specifies the format of the entered parameters. This parameter can be one of the following values:</li> </ul> |

- RTC\_FORMAT\_BIN: Binary data format
  - RTC\_FORMAT\_BCD: BCD data format
- Return values
- **HAL:** status

### **HAL\_RTC\_SetAlarm\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_SetAlarm_IT<br/>(RTC_HandleTypeDef * hrtc, RTC_AlarmTypeDef * sAlarm,<br/>uint32_t Format)</b>                                                                                                                                                                                                                                                                      |
| Function description | Set the specified RTC Alarm with Interrupt.                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>sAlarm:</b> Pointer to Alarm structure</li> <li>• <b>Format:</b> Specifies the format of the entered parameters. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– RTC_FORMAT_BIN: Binary data format</li> <li>– RTC_FORMAT_BCD: BCD data format</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• The Alarm register can only be written when the corresponding Alarm is disabled (Use the HAL_RTC_DeactivateAlarm()).</li> <li>• The HAL_RTC_SetTime() must be called before enabling the Alarm feature.</li> </ul>                                                                                                                                      |

### **HAL\_RTC\_DeactivateAlarm**

|                      |                                                                                                                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_DeactivateAlarm<br/>(RTC_HandleTypeDef * hrtc, uint32_t Alarm)</b>                                                                                                                                                                                        |
| Function description | Deactivate the specified RTC Alarm.                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>Alarm:</b> Specifies the Alarm. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– RTC_ALARM_A : AlarmA</li> <li>– RTC_ALARM_B : AlarmB</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                 |

### **HAL\_RTC\_GetAlarm**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_GetAlarm<br/>(RTC_HandleTypeDef * hrtc, RTC_AlarmTypeDef * sAlarm,<br/>uint32_t Alarm, uint32_t Format)</b>                                                                                                                                                                                                                                                                                |
| Function description | Get the RTC Alarm value and masks.                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>sAlarm:</b> Pointer to Date structure</li> <li>• <b>Alarm:</b> Specifies the Alarm. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– RTC_ALARM_A: AlarmA</li> <li>– RTC_ALARM_B: AlarmB</li> </ul> </li> <li>• <b>Format:</b> Specifies the format of the entered parameters. This</li> </ul> |

parameter can be one of the following values:

- RTC\_FORMAT\_BIN: Binary data format
- RTC\_FORMAT\_BCD: BCD data format

#### Return values

- **HAL:** status

### **HAL\_RTC\_AlarmIRQHandler**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTC_AlarmIRQHandler (RTC_HandleTypeDef * hrtc)</b>              |
| Function description | Handle Alarm interrupt request.                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_RTC\_PollForAlarmAEvent**

|                      |                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_PollForAlarmAEvent (RTC_HandleTypeDef * hrtc, uint32_t Timeout)</b>                        |
| Function description | Handle AlarmA Polling request.                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                  |

### **HAL\_RTC\_AlarmAEventCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTC_AlarmAEventCallback (RTC_HandleTypeDef * hrtc)</b>          |
| Function description | Alarm A callback.                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_RTC\_WaitForSynchro**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTC_WaitForSynchro (RTC_HandleTypeDef * hrtc)</b>  |
| Function description | @addtogroup RTC_Exported_Functions_Group4 Peripheral Control functions      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>      |

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes | <ul style="list-style-type: none"> <li>• The RTC Resynchronization mode is write protected, use the <code>_HAL_RTC_WRITEPROTECTION_DISABLE()</code> before calling this function.</li> <li>• To read the calendar through the shadow registers after Calendar initialization, calendar update or after wakeup from low power modes the software must first clear the RSF flag. The software must then wait until it is set again before reading the calendar, which means that the calendar registers have been correctly copied into the RTC_TR and RTC_DR shadow</li> </ul> |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

registers.

### **HAL\_RTC\_GetState**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_RTCStateTypeDef HAL_RTC_GetState<br/>(RTC_HandleTypeDef * hrtc)</b>  |
| Function description | @addtogroup RTC_Exported_Functions_Group5 Peripheral State functions        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>       |

### **RTC\_EnterInitMode**

|                      |                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef RTC_EnterInitMode<br/>(RTC_HandleTypeDef * hrtc)</b>                                                                                                                                                                                 |
| Function description | @addtogroup RTC_Private_Functions RTC Private Functions                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>                                                                                                                                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– HAL_OK : RTC is in Init mode</li> <li>– HAL_TIMEOUT : RTC is not in Init mode and in Timeout</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• The RTC Initialization mode is write protected, use the <code>__HAL_RTC_WRITEPROTECTION_DISABLE()</code> before calling this function.</li> </ul>                                                                |

### **RTC\_ByteToBcd2**

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <b>uint8_t RTC_ByteToBcd2 (uint8_t Value)</b>                                          |
| Function description | Convert a 2 digit decimal to BCD format.                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Value:</b> Byte to be converted</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Converted:</b> byte</li> </ul>             |

### **RTC\_Bcd2ToByte**

|                      |                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------|
| Function name        | <b>uint8_t RTC_Bcd2ToByte (uint8_t Value)</b>                                               |
| Function description | Convert from 2 digit BCD to Binary.                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Value:</b> BCD value to be converted</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Converted:</b> word</li> </ul>                  |

## **42.3 RTC Firmware driver defines**

### **42.3.1 RTC**

**RTC AlarmDateWeekDay Definitions**

RTC\_ALARMDATEWEEKDAYSEL\_DATE  
RTC\_ALARMDATEWEEKDAYSEL\_WEEKDAY

***RTC AlarmMask Definitions***

RTC\_ALARMMASK\_NONE  
 RTC\_ALARMMASK\_DATEWEEKDAY  
 RTC\_ALARMMASK\_HOURS  
 RTC\_ALARMMASK\_MINUTES  
 RTC\_ALARMMASK\_SECONDS  
 RTC\_ALARMMASK\_ALL

***RTC Alarms Definitions***

RTC\_ALARM\_A  
 RTC\_ALARM\_B

***RTC Alarm Sub Seconds Masks Definitions***

|                                |                                                                                 |
|--------------------------------|---------------------------------------------------------------------------------|
| RTC_ALARMSUBSECONDMASK_ALL     | All Alarm SS fields are masked. There is no comparison on sub seconds for Alarm |
| RTC_ALARMSUBSECONDMASK_SS14_1  | SS[14:1] are ignored in Alarm comparison.<br>Only SS[0] is compared.            |
| RTC_ALARMSUBSECONDMASK_SS14_2  | SS[14:2] are ignored in Alarm comparison.<br>Only SS[1:0] are compared          |
| RTC_ALARMSUBSECONDMASK_SS14_3  | SS[14:3] are ignored in Alarm comparison.<br>Only SS[2:0] are compared          |
| RTC_ALARMSUBSECONDMASK_SS14_4  | SS[14:4] are ignored in Alarm comparison.<br>Only SS[3:0] are compared          |
| RTC_ALARMSUBSECONDMASK_SS14_5  | SS[14:5] are ignored in Alarm comparison.<br>Only SS[4:0] are compared          |
| RTC_ALARMSUBSECONDMASK_SS14_6  | SS[14:6] are ignored in Alarm comparison.<br>Only SS[5:0] are compared          |
| RTC_ALARMSUBSECONDMASK_SS14_7  | SS[14:7] are ignored in Alarm comparison.<br>Only SS[6:0] are compared          |
| RTC_ALARMSUBSECONDMASK_SS14_8  | SS[14:8] are ignored in Alarm comparison.<br>Only SS[7:0] are compared          |
| RTC_ALARMSUBSECONDMASK_SS14_9  | SS[14:9] are ignored in Alarm comparison.<br>Only SS[8:0] are compared          |
| RTC_ALARMSUBSECONDMASK_SS14_10 | SS[14:10] are ignored in Alarm comparison.<br>Only SS[9:0] are compared         |
| RTC_ALARMSUBSECONDMASK_SS14_11 | SS[14:11] are ignored in Alarm comparison.<br>Only SS[10:0] are compared        |
| RTC_ALARMSUBSECONDMASK_SS14_12 | SS[14:12] are ignored in Alarm comparison.<br>Only SS[11:0] are compared        |
| RTC_ALARMSUBSECONDMASK_SS14_13 | SS[14:13] are ignored in Alarm comparison.<br>Only SS[12:0] are compared        |
| RTC_ALARMSUBSECONDMASK_SS14    | SS[14] is don't care in Alarm comparison.<br>Only SS[13:0] are compared         |

`RTC_ALARMSUBSECONDMASK_NONE` SS[14:0] are compared and must match to activate alarm.

***RTC AM PM Definitions***

`RTC_HOURFORMAT12_AM`

`RTC_HOURFORMAT12_PM`

***RTC DayLightSaving Definitions***

`RTC_DAYLIGHTSAVING_NONE`

`RTC_DAYLIGHTSAVING_SUB1H`

`RTC_DAYLIGHTSAVING_ADD1H`

***RTC Exported Macros***

`_HAL_RTC_RESET_HANDLE_STATE`

**Description:**

- Reset RTC handle state.

**Parameters:**

- `_HANDLE_`: RTC handle.

**Return value:**

- None

`_HAL_RTC_WRITEPROTECTION_DISABLE`

**Description:**

- Disable the write protection for RTC registers.

**Parameters:**

- `_HANDLE_`: specifies the RTC handle.

**Return value:**

- None

`_HAL_RTC_WRITEPROTECTION_ENABLE`

**Description:**

- Enable the write protection for RTC registers.

**Parameters:**

- `_HANDLE_`: specifies the RTC handle.

**Return value:**

- None

`_HAL_RTC_ALARMA_ENABLE`

**Description:**

- Enable the RTC ALARMA peripheral.

**Parameters:**

- `_HANDLE_`: specifies the RTC handle.

`__HAL_RTC_ALARMA_DISABLE`

**Return value:**

- None

**Description:**

- Disable the RTC ALARMA peripheral.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_ALARM_B_ENABLE`

**Description:**

- Enable the RTC ALARM\_B peripheral.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_ALARM_B_DISABLE`

**Description:**

- Disable the RTC ALARM\_B peripheral.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_ALARM_ENABLE_IT`

**Description:**

- Enable the RTC Alarm interrupt.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.
- `__INTERRUPT__`: specifies the RTC Alarm interrupt sources to be enabled or disabled. This parameter can be any combination of the following values:
  - `RTC_IT_ALRA`: Alarm A interrupt
  - `RTC_IT_ALRB`: Alarm B interrupt

**Return value:**

- None

#### \_HAL\_RTC\_ALARM\_DISABLE\_IT

- Disable the RTC Alarm interrupt.

#### **Parameters:**

- \_HANDLE\_: specifies the RTC handle.
- \_INTERRUPT\_: specifies the RTC Alarm interrupt sources to be enabled or disabled. This parameter can be any combination of the following values:
  - RTC\_IT\_ALRA: Alarm A interrupt
  - RTC\_IT\_ALRB: Alarm B interrupt

#### **Return value:**

- None

#### \_HAL\_RTC\_ALARM\_GET\_IT

- Check whether the specified RTC Alarm interrupt has occurred or not.

#### **Parameters:**

- \_HANDLE\_: specifies the RTC handle.
- \_INTERRUPT\_: specifies the RTC Alarm interrupt to check. This parameter can be:
  - RTC\_IT\_ALRA: Alarm A interrupt
  - RTC\_IT\_ALRB: Alarm B interrupt

#### **Return value:**

- None

#### \_HAL\_RTC\_ALARM\_GET\_IT\_SOURCE

- Check whether the specified RTC Alarm interrupt has been enabled or not.

#### **Parameters:**

- \_HANDLE\_: specifies the RTC handle.
- \_INTERRUPT\_: specifies the RTC Alarm interrupt sources to check. This parameter can be:
  - RTC\_IT\_ALRA: Alarm A interrupt
  - RTC\_IT\_ALRB: Alarm B interrupt

---

interrupt

**Return value:**

- None

`__HAL_RTC_ALARM_GET_FLAG`

**Description:**

- Get the selected RTC Alarm's flag status.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.
- `__FLAG__`: specifies the RTC Alarm Flag sources to check. This parameter can be:
  - `RTC_FLAG_ALRAF`
  - `RTC_FLAG_ALRBF`
  - `RTC_FLAG_ALRAWF`
  - `RTC_FLAG_ALRBWF`

**Return value:**

- None

`__HAL_RTC_ALARM_CLEAR_FLAG`

**Description:**

- Clear the RTC Alarm's pending flags.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.
- `__FLAG__`: specifies the RTC Alarm Flag sources to clear. This parameter can be:
  - `RTC_FLAG_ALRAF`
  - `RTC_FLAG_ALRBF`

**Return value:**

- None

`__HAL_RTC_ALARM_EXTI_ENABLE_IT`

**Description:**

- Enable interrupt on the RTC Alarm associated Exti line.

**Return value:**

- None

`__HAL_RTC_ALARM_EXTI_DISABLE_IT`

**Description:**

- Disable interrupt on the RTC Alarm associated Exti line.

**Return value:**

- None

`__HAL_RTC_ALARM_EXTI_ENABLE_EVENT`

**Description:**

- Enable event on the RTC Alarm associated Exti line.

**Return value:**

- None.

`__HAL_RTC_ALARM_EXTI_DISABLE_EVENT`

**Description:**

- Disable event on the RTC Alarm associated Exti line.

**Return value:**

- None.

`__HAL_RTC_ALARM_EXTI_ENABLE_FALLING_EDGE`

**Description:**

- Enable falling edge trigger on the RTC Alarm associated Exti line.

**Return value:**

- None.

`__HAL_RTC_ALARM_EXTI_DISABLE_FALLING_EDGE`

**Description:**

- Disable falling edge trigger on the RTC Alarm associated Exti line.

**Return value:**

- None.

`__HAL_RTC_ALARM_EXTI_ENABLE_RISING_EDGE`

**Description:**

- Enable rising edge trigger on the RTC Alarm associated Exti line.

**Return value:**

- None.

`__HAL_RTC_ALARM_EXTI_DISABLE_RISING_EDGE`

**Description:**

- Disable rising edge trigger on the RTC Alarm associated Exti line.

**Return value:**

- None.

`__HAL_RTC_ALARM_EXTI_ENABLE_RISING_FALLING_EDGE`

**Description:**

- Enable rising & falling edge trigger on the RTC Alarm associated Exti line.

**Return value:**

- None.

---

`__HAL_RTC_ALARM_EXTI_DISABLE_RISING_EDGE`

**Description:**

- Disable rising & falling edge trigger on the RTC Alarm associated Exti line.

**Return value:**

- None.

`__HAL_RTC_ALARM_EXTI_GET_FLAG`

**Description:**

- Check whether the RTC Alarm associated Exti line interrupt flag is set or not.

**Return value:**

- Line: Status.

`__HAL_RTC_ALARM_EXTI_CLEAR_FLAG`

**Description:**

- Clear the RTC Alarm associated Exti line flag.

**Return value:**

- None.

`__HAL_RTC_ALARM_EXTI_GENERATE_SWIT`

**Description:**

- Generate a Software interrupt on RTC Alarm associated Exti line.

**Return value:**

- None.

***RTC Flags Definitions***

`RTC_FLAG_RECALPF`

`RTC_FLAG_TAMP3F`

`RTC_FLAG_TAMP2F`

`RTC_FLAG_TAMP1F`

`RTC_FLAG_TSOVF`

`RTC_FLAG_TSF`

`RTC_FLAG_WUTF`

`RTC_FLAG_ALRBF`

`RTC_FLAG_ALRAF`

`RTC_FLAG_INITF`

`RTC_FLAG_RSF`

`RTC_FLAG_INITS`

`RTC_FLAG_SHPF`

`RTC_FLAG_WUTWF`

`RTC_FLAG_ALRBWF`

RTC\_FLAG\_ALRAWF

**RTC Hour Formats**

RTC\_HOURFORMAT\_24

RTC\_HOURFORMAT\_12

**RTC Input parameter format definitions**

RTC\_FORMAT\_BIN

RTC\_FORMAT\_BCD

**RTC Interrupts Definitions**

RTC\_IT\_TS

RTC\_IT\_WUT

RTC\_IT\_ALRB

RTC\_IT\_ALRA

RTC\_IT\_TAMP

RTC\_IT\_TAMP1

RTC\_IT\_TAMP2

RTC\_IT\_TAMP3

**RTC Private macros to check input parameters**

IS\_RTC\_HOUR\_FORMAT

IS\_RTC\_OUTPUT\_POL

IS\_RTC\_OUTPUT\_TYPE

IS\_RTC\_HOUR12

IS\_RTC\_HOUR24

IS\_RTC\_ASYNCH\_PREDIV

IS\_RTC\_SYNCH\_PREDIV

IS\_RTC\_MINUTES

IS\_RTC\_SECONDS

IS\_RTC\_HOURFORMAT12

IS\_RTC\_DAYLIGHT\_SAVING

IS\_RTC\_STORE\_OPERATION

IS\_RTC\_FORMAT

IS\_RTC\_YEAR

IS\_RTC\_MONTH

IS\_RTC\_DATE

IS\_RTC\_WEEKDAY

IS\_RTC\_ALARM\_DATE\_WEEKDAY\_DATE

IS\_RTC\_ALARM\_DATE\_WEEKDAY\_WEEKDAY

IS\_RTC\_ALARM\_DATE\_WEEKDAY\_SEL  
IS\_RTC\_ALARM\_MASK  
IS\_RTC\_ALARM  
IS\_RTC\_ALARM\_SUB\_SECOND\_VALUE  
IS\_RTC\_ALARM\_SUB\_SECOND\_MASK

***RTC Month Date Definitions***

RTC\_MONTH\_JANUARY  
RTC\_MONTH\_FEBRUARY  
RTC\_MONTH\_MARCH  
RTC\_MONTH\_APRIIL  
RTC\_MONTH\_MAY  
RTC\_MONTH\_JUNE  
RTC\_MONTH\_JULY  
RTC\_MONTH\_AUGUST  
RTC\_MONTH\_SEPTEMBER  
RTC\_MONTH\_OCTOBER  
RTC\_MONTH\_NOVEMBER  
RTC\_MONTH\_DECEMBER

***RTC Output Polarity Definitions***

RTC\_OUTPUT\_POLARITY\_HIGH  
RTC\_OUTPUT\_POLARITY\_LOW

***RTC Output Type ALARM OUT***

RTC\_OUTPUT\_TYPE\_OPENDRAIN  
RTC\_OUTPUT\_TYPE\_PUSH\_PULL

***RTC StoreOperation Definitions***

RTC\_STOREOPERATION\_RESET  
RTC\_STOREOPERATION\_SET

***RTC WeekDay Definitions***

RTC\_WEEKDAY\_MONDAY  
RTC\_WEEKDAY\_TUESDAY  
RTC\_WEEKDAY\_WEDNESDAY  
RTC\_WEEKDAY\_THURSDAY  
RTC\_WEEKDAY\_FRIDAY  
RTC\_WEEKDAY\_SATURDAY  
RTC\_WEEKDAY\_SUNDAY

## 43 HAL RTC Extension Driver

### 43.1 RTCEEx Firmware driver registers structures

#### 43.1.1 RTC\_TamperTypeDef

##### Data Fields

- *uint32\_t Tamper*
- *uint32\_t Trigger*
- *uint32\_t Filter*
- *uint32\_t SamplingFrequency*
- *uint32\_t PrechargeDuration*
- *uint32\_t TamperPullUp*
- *uint32\_t TimeStampOnTamperDetection*

##### Field Documentation

- *uint32\_t RTC\_TamperTypeDef::Tamper*  
Specifies the Tamper Pin. This parameter can be a value of  
[\*RTCEEx\\_Tamper\\_Pins\\_Definitions\*](#)
- *uint32\_t RTC\_TamperTypeDef::Trigger*  
Specifies the Tamper Trigger. This parameter can be a value of  
[\*RTCEEx\\_Tamper\\_Trigger\\_Definitions\*](#)
- *uint32\_t RTC\_TamperTypeDef::Filter*  
Specifies the RTC Filter Tamper. This parameter can be a value of  
[\*RTCEEx\\_Tamper\\_Filter\\_Definitions\*](#)
- *uint32\_t RTC\_TamperTypeDef::SamplingFrequency*  
Specifies the sampling frequency. This parameter can be a value of  
[\*RTCEEx\\_Tamper\\_Sampling\\_Frequencies\\_Definitions\*](#)
- *uint32\_t RTC\_TamperTypeDef::PrechargeDuration*  
Specifies the Precharge Duration . This parameter can be a value of  
[\*RTCEEx\\_Tamper\\_Pin\\_Precharge\\_Duration\\_Definitions\*](#)
- *uint32\_t RTC\_TamperTypeDef::TamperPullUp*  
Specifies the Tamper PullUp . This parameter can be a value of  
[\*RTCEEx\\_Tamper\\_Pull\\_UP\\_Definitions\*](#)
- *uint32\_t RTC\_TamperTypeDef::TimeStampOnTamperDetection*  
Specifies the TimeStampOnTamperDetection. This parameter can be a value of  
[\*RTCEEx\\_Tamper\\_TimeStampOnTamperDetection\\_Definitions\*](#)

### 43.2 RTCEEx Firmware driver API description

#### 43.2.1 How to use this driver

- Enable the RTC domain access.
- Configure the RTC Prescaler (Asynchronous and Synchronous) and RTC hour format using the HAL\_RTC\_Init() function.

##### RTC Wakeup configuration

- To configure the RTC Wakeup Clock source and Counter use the HAL\_RTCEEx\_SetWakeUpTimer() function. You can also configure the RTC Wakeup timer with interrupt mode using the HAL\_RTCEEx\_SetWakeUpTimer\_IT() function.

- To read the RTC WakeUp Counter register, use the HAL\_RTCEEx\_SetWakeUpTimer() function.

### TimeStamp configuration

- Configure the RTC\_AF trigger and enable the RTC TimeStamp using the HAL\_RTCEEx\_SetTimeStamp() function. You can also configure the RTC TimeStamp with interrupt mode using the HAL\_RTCEEx\_SetTimeStamp\_IT() function.
- To read the RTC TimeStamp Time and Date register, use the HAL\_RTCEEx\_GetTimeStamp() function.
- The TIMESTAMP alternate function is mapped to RTC\_AF1 (PC13U).

### Tamper configuration

- Enable the RTC Tamper and configure the Tamper filter count, trigger Edge or Level according to the Tamper filter (if equal to 0 Edge else Level) value, sampling frequency, precharge or discharge and Pull-UP using the HAL\_RTCEEx\_SetTamper() function. You can configure RTC Tamper with interrupt mode using HAL\_RTCEEx\_SetTamper\_IT() function.
- The TAMPER1 alternate function is mapped to RTC\_AF1 (PC13U).

### Backup Data Registers configuration

- To write to the RTC Backup Data registers, use the HAL\_RTCEEx\_BKUPWrite() function.
- To read the RTC Backup Data registers, use the HAL\_RTCEEx\_BKUPRead() function.

## 43.2.2 RTC TimeStamp and Tamper functions

This section provides functions allowing to configure TimeStamp feature

This section contains the following APIs:

- [`HAL\_RTCEEx\_SetTimeStamp\(\)`](#)
- [`HAL\_RTCEEx\_SetTimeStamp\_IT\(\)`](#)
- [`HAL\_RTCEEx\_DeactivateTimeStamp\(\)`](#)
- [`HAL\_RTCEEx\_GetTimeStamp\(\)`](#)
- [`HAL\_RTCEEx\_SetTamper\(\)`](#)
- [`HAL\_RTCEEx\_SetTamper\_IT\(\)`](#)
- [`HAL\_RTCEEx\_DeactivateTamper\(\)`](#)
- [`HAL\_RTCEEx\_TamperTimeStampIRQHandler\(\)`](#)
- [`HAL\_RTCEEx\_TimeStampEventCallback\(\)`](#)
- [`HAL\_RTCEEx\_Tamper1EventCallback\(\)`](#)
- [`HAL\_RTCEEx\_Tamper2EventCallback\(\)`](#)
- [`HAL\_RTCEEx\_Tamper3EventCallback\(\)`](#)
- [`HAL\_RTCEEx\_PollForTimeStampEvent\(\)`](#)
- [`HAL\_RTCEEx\_PollForTamper1Event\(\)`](#)
- [`HAL\_RTCEEx\_PollForTamper2Event\(\)`](#)
- [`HAL\_RTCEEx\_PollForTamper3Event\(\)`](#)

## 43.2.3 RTC Wake-up functions

This section provides functions allowing to configure Wake-up feature

This section contains the following APIs:

- [`HAL\_RTCEEx\_SetWakeUpTimer\(\)`](#)
- [`HAL\_RTCEEx\_SetWakeUpTimer\_IT\(\)`](#)

- [\*HAL\\_RTCEx\\_DeactivateWakeUpTimer\(\)\*](#)
- [\*HAL\\_RTCEx\\_GetWakeUpTimer\(\)\*](#)
- [\*HAL\\_RTCEx\\_WakeUpTimerIRQHandler\(\)\*](#)
- [\*HAL\\_RTCEx\\_WakeUpTimerEventCallback\(\)\*](#)
- [\*HAL\\_RTCEx\\_PollForWakeUpTimerEvent\(\)\*](#)

#### 43.2.4 Extended Peripheral Control functions

This subsection provides functions allowing to

- Write a data in a specified RTC Backup data register
- Read a data in a specified RTC Backup data register
- Set the Coarse calibration parameters.
- Deactivate the Coarse calibration parameters
- Set the Smooth calibration parameters.
- Configure the Synchronization Shift Control Settings.
- Configure the Calibration Pinout (RTC\_CALIB) Selection (1Hz or 512Hz).
- Deactivate the Calibration Pinout (RTC\_CALIB) Selection (1Hz or 512Hz).
- Enable the RTC reference clock detection.
- Disable the RTC reference clock detection.
- Enable the Bypass Shadow feature.
- Disable the Bypass Shadow feature.

This section contains the following APIs:

- [\*HAL\\_RTCEx\\_BKUPWrite\(\)\*](#)
- [\*HAL\\_RTCEx\\_BKUPRead\(\)\*](#)
- [\*HAL\\_RTCEx\\_SetSmoothCalib\(\)\*](#)
- [\*HAL\\_RTCEx\\_SetSynchroShift\(\)\*](#)
- [\*HAL\\_RTCEx\\_SetCalibrationOutPut\(\)\*](#)
- [\*HAL\\_RTCEx\\_DeactivateCalibrationOutPut\(\)\*](#)
- [\*HAL\\_RTCEx\\_SetRefClock\(\)\*](#)
- [\*HAL\\_RTCEx\\_DeactivateRefClock\(\)\*](#)
- [\*HAL\\_RTCEx\\_EnableBypassShadow\(\)\*](#)
- [\*HAL\\_RTCEx\\_DisableBypassShadow\(\)\*](#)

#### 43.2.5 Extended features functions

This section provides functions allowing to:

- RTC Alram B callback
- RTC Poll for Alarm B request

This section contains the following APIs:

- [\*HAL\\_RTCEx\\_AlarmBEventCallback\(\)\*](#)
- [\*HAL\\_RTCEx\\_PollForAlarmBEvent\(\)\*](#)

#### 43.2.6 Detailed description of functions

##### **HAL\_RTCEx\_SetTimeStamp**

Function name      **HAL\_StatusTypeDef HAL\_RTCEx\_SetTimeStamp  
(RTC\_HandleTypeDef \* hrtc, uint32\_t TimeStampEdge,  
uint32\_t RTC\_TimeStampPin)**

Function description      SetTimeStamp.

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>TimeStampEdge:</b> Specifies the pin edge on which the TimeStamp is activated. This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– RTC_TIMESTAMPEDGE_RISING: the Time stamp event occurs on the rising edge of the related pin.</li> <li>– RTC_TIMESTAMPEDGE_FALLING: the Time stamp event occurs on the falling edge of the related pin.</li> </ul> </li> <li>• <b>RTC_TimeStampPin:</b> specifies the RTC TimeStamp Pin. This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– RTC_TIMESTAMPPIN_DEFAULT: PC13 is selected as RTC TimeStamp Pin.</li> </ul> </li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes         | <ul style="list-style-type: none"> <li>• This API must be called before enabling the TimeStamp feature.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

### HAL\_RTCEx\_SetTimeStamp\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_SetTimeStamp_IT<br/>(RTC_HandleTypeDef * hrtc, uint32_t TimeStampEdge,<br/>uint32_t RTC_TimeStampPin)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Set TimeStamp with Interrupt.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>TimeStampEdge:</b> Specifies the pin edge on which the TimeStamp is activated. This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– RTC_TIMESTAMPEDGE_RISING: the Time stamp event occurs on the rising edge of the related pin.</li> <li>– RTC_TIMESTAMPEDGE_FALLING: the Time stamp event occurs on the falling edge of the related pin.</li> </ul> </li> <li>• <b>RTC_TimeStampPin:</b> Specifies the RTC TimeStamp Pin. This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– RTC_TIMESTAMPPIN_DEFAULT: PC13 is selected as RTC TimeStamp Pin.</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• This API must be called before enabling the TimeStamp feature.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

### HAL\_RTCEx\_DeactivateTimeStamp

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_DeactivateTimeStamp<br/>(RTC_HandleTypeDef * hrtc)</b> |
| Function description | Deactivate TimeStamp.                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                |

**HAL\_RTCEx\_GetTimeStamp**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_GetTimeStamp<br/>(RTC_HandleTypeDef * hrtc, RTC_TimeTypeDef *<br/>sTimeStamp, RTC_DateTypeDef * sTimeStampDate, uint32_t<br/>Format)</b>                                                                                                                                                                                                                                                                                         |
| Function description | Get the RTC TimeStamp value.                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>sTimeStamp:</b> Pointer to Time structure</li> <li>• <b>sTimeStampDate:</b> Pointer to Date structure</li> <li>• <b>Format:</b> specifies the format of the entered parameters. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– RTC_FORMAT_BIN: Binary data format</li> <li>– RTC_FORMAT_BCD: BCD data format</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                          |

**HAL\_RTCEx\_SetTamper**

|                      |                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_SetTamper<br/>(RTC_HandleTypeDef * hrtc, RTC_TamperTypeDef * sTamper)</b>                            |
| Function description | Set Tamper.                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>sTamper:</b> Pointer to Tamper Structure.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                              |

**Notes**

- By calling this API we disable the tamper interrupt for all tampers.

**HAL\_RTCEx\_SetTamper\_IT**

|                      |                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_SetTamper_IT<br/>(RTC_HandleTypeDef * hrtc, RTC_TamperTypeDef * sTamper)</b>                   |
| Function description | Set Tamper with interrupt.                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>sTamper:</b> Pointer to RTC Tamper.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                        |

**Notes**

- By calling this API we force the tamper interrupt for all tampers.

**HAL\_RTCEx\_DeactivateTamper**

|                      |                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_DeactivateTamper<br/>(RTC_HandleTypeDef * hrtc, uint32_t Tamper)</b>                                                                                                                 |
| Function description | Deactivate Tamper.                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>Tamper:</b> Selected tamper pin. This parameter can be any combination of RTC_TAMPER_1, RTC_TAMPER_2 and RTC_TAMPER_3 (*)</li> </ul> |

---

|               |                                                                                                     |
|---------------|-----------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                              |
| Notes         | <ul style="list-style-type: none"> <li>• (*) RTC_TAMPER_3 not present on all the devices</li> </ul> |

### **HAL\_RTCEx\_TamperTimeStampIRQHandler**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTCEx_TamperTimeStampIRQHandler<br/>(RTC_HandleTypeDef * hrtc)</b> |
| Function description | Handle TimeStamp interrupt request.                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                |

### **HAL\_RTCEx\_Tamper1EventCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTCEx_Tamper1EventCallback<br/>(RTC_HandleTypeDef * hrtc)</b>   |
| Function description | Tamper 1 callback.                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_RTCEx\_Tamper2EventCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTCEx_Tamper2EventCallback<br/>(RTC_HandleTypeDef * hrtc)</b>   |
| Function description | Tamper 2 callback.                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_RTCEx\_Tamper3EventCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTCEx_Tamper3EventCallback<br/>(RTC_HandleTypeDef * hrtc)</b>   |
| Function description | Tamper 3 callback.                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_RTCEx\_TimeStampEventCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTCEx_TimeStampEventCallback<br/>(RTC_HandleTypeDef * hrtc)</b> |
| Function description | TimeStamp callback.                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_RTCEx\_PollForTimeStampEvent**

|                      |                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_PollForTimeStampEvent<br/>(RTC_HandleTypeDef * hrtc, uint32_t Timeout)</b>            |
| Function description | Handle TimeStamp polling request.                                                                                    |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hrtc:</b> RTC handle</li><li>• <b>Timeout:</b> Timeout duration</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                 |

**HAL\_RTCEx\_PollForTamper1Event**

|                      |                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_PollForTamper1Event<br/>(RTC_HandleTypeDef * hrtc, uint32_t Timeout)</b>              |
| Function description | Handle Tamper 1 Polling.                                                                                             |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hrtc:</b> RTC handle</li><li>• <b>Timeout:</b> Timeout duration</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                 |

**HAL\_RTCEx\_PollForTamper2Event**

|                      |                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_PollForTamper2Event<br/>(RTC_HandleTypeDef * hrtc, uint32_t Timeout)</b>              |
| Function description | Handle Tamper 2 Polling.                                                                                             |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hrtc:</b> RTC handle</li><li>• <b>Timeout:</b> Timeout duration</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                 |

**HAL\_RTCEx\_PollForTamper3Event**

|                      |                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_PollForTamper3Event<br/>(RTC_HandleTypeDef * hrtc, uint32_t Timeout)</b>              |
| Function description | Handle Tamper 3 Polling.                                                                                             |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hrtc:</b> RTC handle</li><li>• <b>Timeout:</b> Timeout duration</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                 |

**HAL\_RTCEx\_SetWakeUpTimer**

|                      |                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_SetWakeUpTimer<br/>(RTC_HandleTypeDef * hrtc, uint32_t WakeUpCounter,<br/>uint32_t WakeUpClock)</b>                                    |
| Function description | Set wake up timer.                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hrtc:</b> RTC handle</li><li>• <b>WakeUpCounter:</b> Wake up counter</li><li>• <b>WakeUpClock:</b> Wake up clock</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                  |

**HAL\_RTCEx\_SetWakeUpTimer\_IT**

|                      |                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_SetWakeUpTimer_IT<br/>(RTC_HandleTypeDef * hrtc, uint32_t WakeUpCounter,<br/>uint32_t WakeUpClock)</b>                                     |
| Function description | Set wake up timer with interrupt.                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>WakeUpCounter:</b> Wake up counter</li> <li>• <b>WakeUpClock:</b> Wake up clock</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                    |

**HAL\_RTCEx\_DeactivateWakeUpTimer**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_RTCEx_DeactivateWakeUpTimer<br/>(RTC_HandleTypeDef * hrtc)</b> |
| Function description | Deactivate wake up timer counter.                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>         |

**HAL\_RTCEx\_GetWakeUpTimer**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_RTCEx_GetWakeUpTimer (RTC_HandleTypeDef<br/>* hrtc)</b>     |
| Function description | Get wake up timer counter.                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Counter:</b> value</li> </ul>   |

**HAL\_RTCEx\_WakeUpTimerIRQHandler**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTCEx_WakeUpTimerIRQHandler<br/>(RTC_HandleTypeDef * hrtc)</b>  |
| Function description | Handle Wake Up Timer interrupt request.                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_RTCEx\_WakeUpTimerEventCallback**

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTCEx_WakeUpTimerEventCallback<br/>(RTC_HandleTypeDef * hrtc)</b> |
| Function description | Wake Up Timer callback.                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>               |

**HAL\_RTCEx\_PollForWakeUpTimerEvent**

|                      |                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_PollForWakeUpTimerEvent (RTC_HandleTypeDef * hrtc, uint32_t Timeout)</b>                 |
| Function description | Handle Wake Up Timer Polling.                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                  |

**HAL\_RTCEx\_BKUPWrite**

|                      |                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_RTCEx_BKUPWrite (RTC_HandleTypeDef * hrtc, uint32_t BackupRegister, uint32_t Data)</b>                                                                                                                                                                                                                           |
| Function description | Write a data in a specified RTC Backup data register.                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>BackupRegister:</b> RTC Backup data Register number. This parameter can be: RTC_BKP_DRx where x can be from 0 to 19 to specify the register.</li> <li>• <b>Data:</b> Data to be written in the specified RTC Backup data register.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                              |

**HAL\_RTCEx\_BKUPRead**

|                      |                                                                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_RTCEx_BKUPRead (RTC_HandleTypeDef * hrtc, uint32_t BackupRegister)</b>                                                                                                                                                 |
| Function description | Reads data from the specified RTC Backup data Register.                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>BackupRegister:</b> RTC Backup data Register number. This parameter can be: RTC_BKP_DRx where x can be from 0 to 19 to specify the register.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Read:</b> value</li> </ul>                                                                                                                                                                 |

**HAL\_RTCEx\_SetSmoothCalib**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_SetSmoothCalib (RTC_HandleTypeDef * hrtc, uint32_t SmoothCalibPeriod, uint32_t SmoothCalibPlusPulses, uint32_t SmoothCalibMinusPulsesValue)</b>                                                                                                                                                                                                                                                                                                                  |
| Function description | Set the Smooth calibration parameters.                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>SmoothCalibPeriod:</b> Select the Smooth Calibration Period. This parameter can be one of the following values : <ul style="list-style-type: none"> <li>– RTC_SMOOTHCALIB_PERIOD_32SEC: The smooth calibration period is 32s.</li> <li>– RTC_SMOOTHCALIB_PERIOD_16SEC: The smooth calibration period is 16s.</li> <li>– RTC_SMOOTHCALIB_PERIOD_8SEC: The smooth calibration period is 8s.</li> </ul> </li> </ul> |

- **SmoothCalibPlusPulses:** Select to Set or reset the CALP bit. This parameter can be one of the following values:
  - RTC\_SMOOTHCALIB\_PLUSPULSES\_SET: Add one RTCCLK pulse every 2\*11 pulses.
  - RTC\_SMOOTHCALIB\_PLUSPULSES\_RESET: No RTCCLK pulses are added.
- **SmoothCalibMinusPulsesValue:** Select the value of CALM[8:0] bits. This parameter can be one any value from 0 to 0x000001FF.

Return values

- **HAL:** status

Notes

- To deactivate the smooth calibration, the field SmoothCalibPlusPulses must be equal to SMOOTHCALIB\_PLUSPULSES\_RESET and the field SmoothCalibMinusPulsesValue mut be equal to 0.

### **HAL\_RTCEx\_SetSynchroShift**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_SetSynchroShift<br/>(RTC_HandleTypeDef * hrtc, uint32_t ShiftAdd1S, uint32_t ShiftSubFS)</b>                                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Configure the Synchronization Shift Control Settings.                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>ShiftAdd1S:</b> Select to add or not 1 second to the time calendar. This parameter can be one of the following values :           <ul style="list-style-type: none"> <li>– RTC_SHIFTADD1S_SET: Add one second to the clock calendar.</li> <li>– RTC_SHIFTADD1S_RESET: No effect.</li> </ul> </li> <li>• <b>ShiftSubFS:</b> Select the number of Second Fractions to substitute. This parameter can be one any value from 0 to 0x7FFF.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• When REFCKON is set, firmware must not write to Shift control register.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                     |

### **HAL\_RTCEx\_SetCalibrationOutPut**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_SetCalibrationOutPut<br/>(RTC_HandleTypeDef * hrtc, uint32_t CalibOutput)</b>                                                                                                                                                                                                                                                                                                    |
| Function description | Configure the Calibration Pinout (RTC_CALIB) Selection (1Hz or 512Hz).                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> <li>• <b>CalibOutput:</b> Select the Calibration output Selection . This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– RTC_CALIBOUTPUT_512HZ: A signal has a regular waveform at 512Hz.</li> <li>– RTC_CALIBOUTPUT_1HZ: A signal has a regular waveform at 1Hz.</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                          |

**HAL\_RTCEx\_DeactivateCalibrationOutPut**

|                      |                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_DeactivateCalibrationOutPut<br/>(RTC_HandleTypeDef * hrtc)</b> |
| Function description | Deactivate the Calibration Pinout (RTC_CALIB) Selection (1Hz or 512Hz).                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                        |

**HAL\_RTCEx\_SetRefClock**

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_SetRefClock<br/>(RTC_HandleTypeDef * hrtc)</b> |
| Function description | Enable the RTC reference clock detection.                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>        |

**HAL\_RTCEx\_DeactivateRefClock**

|                      |                                                                                      |
|----------------------|--------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_DeactivateRefClock<br/>(RTC_HandleTypeDef * hrtc)</b> |
| Function description | Disable the RTC reference clock detection.                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>               |

**HAL\_RTCEx\_EnableBypassShadow**

|                      |                                                                                      |
|----------------------|--------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_EnableBypassShadow<br/>(RTC_HandleTypeDef * hrtc)</b> |
| Function description | Enable the Bypass Shadow feature.                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>               |

**Notes**

- When the Bypass Shadow is enabled the calendar value are taken directly from the Calendar counter.

**HAL\_RTCEx\_DisableBypassShadow**

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_RTCEx_DisableBypassShadow<br/>(RTC_HandleTypeDef * hrtc)</b> |
| Function description | Disable the Bypass Shadow feature.                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hrtc:</b> RTC handle</li> </ul>           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                |

**Notes**

- When the Bypass Shadow is enabled the calendar value are taken directly from the Calendar counter.

**HAL\_RTCEx\_AlarmBEventCallback**

Function name      **void HAL\_RTCEx\_AlarmBEventCallback  
(RTC\_HandleTypeDef \* hrtc)**

Function description      Alarm B callback.

Parameters      • **hrtc:** RTC handle

Return values      • **None**

**HAL\_RTCEx\_PollForAlarmBEvent**

Function name      **HAL\_StatusTypeDef HAL\_RTCEx\_PollForAlarmBEvent  
(RTC\_HandleTypeDef \* hrtc, uint32\_t Timeout)**

Function description      This function handles AlarmB Polling request.

Parameters      • **hrtc:** RTC handle  
• **Timeout:** Timeout duration

Return values      • **HAL:** status

## 43.3 RTCEEx Firmware driver defines

### 43.3.1 RTCEEx

*RTC Extended Add 1 Second Parameter Definition*

RTC\_SHIFTADD1S\_RESET

RTC\_SHIFTADD1S\_SET

*RTC Extended Backup Registers Definition*

RTC\_BKP\_DR0

RTC\_BKP\_DR1

RTC\_BKP\_DR2

RTC\_BKP\_DR3

RTC\_BKP\_DR4

RTC\_BKP\_DR5

RTC\_BKP\_DR6

RTC\_BKP\_DR7

RTC\_BKP\_DR8

RTC\_BKP\_DR9

RTC\_BKP\_DR10

RTC\_BKP\_DR11

RTC\_BKP\_DR12

RTC\_BKP\_DR13

RTC\_BKP\_DR14

RTC\_BKP\_DR15

***RTC Extended Calibration***`__HAL_RTC_CALIBRATION_OUTPUT_ENABLE`**Description:**

- Enable the RTC calibration output.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_CALIBRATION_OUTPUT_DISABLE`**Description:**

- Disable the calibration output.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_CLOCKREF_DETECTION_ENABLE`**Description:**

- Enable the clock reference detection.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_CLOCKREF_DETECTION_DISABLE`**Description:**

- Disable the clock reference detection.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_SHIFT_GET_FLAG`**Description:**

- Get the selected RTC shift operation's flag status.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.
- `__FLAG__`: specifies the RTC shift operation Flag is pending

or not. This parameter can be:

- RTC\_FLAG\_SHPF

#### **Return value:**

- None

#### ***RTC Extended Calib Output selection Definition***

RTC\_CALIBOUTPUT\_512HZ

RTC\_CALIBOUTPUT\_1HZ

#### ***Private macros to check input parameters***

IS\_RTC\_OUTPUT

IS\_RTC\_BKP

IS\_TIMESTAMP\_EDGE

IS\_RTC\_TAMPER

IS\_RTC\_TIMESTAMP\_PIN

IS\_RTC\_TAMPER\_TRIGGER

IS\_RTC\_TAMPER\_FILTER

IS\_RTC\_TAMPER\_SAMPLING\_FREQ

IS\_RTC\_TAMPER\_PRECHARGE\_DURATION

IS\_RTC\_TAMPER\_TIMESTAMPTAMPER\_DETECTION

IS\_RTC\_TAMPER\_PULLUP\_STATE

IS\_RTC\_WAKEUP\_CLOCK

IS\_RTC\_WAKEUP\_COUNTER

IS\_RTC\_SMOOTH\_CALIB\_PERIOD

IS\_RTC\_SMOOTH\_CALIB\_PLUS

IS\_RTC\_SMOOTH\_CALIB\_MINUS

IS\_RTC\_SHIFT\_ADD1S

IS\_RTC\_SHIFT\_SUBFS

IS\_RTC\_CALIB\_OUTPUT

#### ***RTC Extended Output Selection Definition***

RTC\_OUTPUT\_DISABLE

RTC\_OUTPUT\_ALARMA

RTC\_OUTPUT\_ALARMB

RTC\_OUTPUT\_WAKEUP

#### ***RTC Extended Smooth calib period Definition***

RTC\_SMOOTHCALIB\_PERIOD\_32SEC If RTCCLK = 32768 Hz, Smooth calibration period is 32s, else 2exp20 RTCCLK seconds

RTC\_SMOOTHCALIB\_PERIOD\_16SEC If RTCCLK = 32768 Hz, Smooth calibration period is 16s, else 2exp19 RTCCLK seconds

`RTC_SMOOTHCALIB_PERIOD_8SEC` If RTCCLK = 32768 Hz, Smooth calibration period is 8s, else  $2^{\text{exp}18}$  RTCCLK seconds

***RTC Extended Smooth calib Plus pulses Definition***

`RTC_SMOOTHCALIB_PLUSPULSES_RESET` The number of RTCCLK pulses substituted during a 32-second window = CALM[8:0]

`RTC_SMOOTHCALIB_PLUSPULSES_SET` The number of RTCCLK pulses added during a X -second window = Y - CALM[8:0] with Y = 512U, 256U, 128 when X = 32U, 16U, 8U

***RTC Extended Tamper***

`__HAL_RTC_TAMPER1_ENABLE`

**Description:**

- Enable the RTC Tamper1 input detection.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_TAMPER1_DISABLE`

**Description:**

- Disable the RTC Tamper1 input detection.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_TAMPER2_ENABLE`

**Description:**

- Enable the RTC Tamper2 input detection.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_TAMPER2_DISABLE`

**Description:**

- Disable the RTC Tamper2 input detection.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

`__HAL_RTC_TAMPER3_ENABLE`**Return value:**

- None

**Description:**

- Enable the RTC Tamper3 input detection.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_TAMPER3_DISABLE`**Description:**

- Disable the RTC Tamper3 input detection.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.

**Return value:**

- None

`__HAL_RTC_TAMPER_ENABLE_IT`**Description:**

- Enable the RTC Tamper interrupt.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.
- `__INTERRUPT__`: specifies the RTC Tamper interrupt sources to be enabled. This parameter can be any combination of the following values:
  - `RTC_IT_TAMP`: Tamper interrupt

**Return value:**

- None

`__HAL_RTC_TAMPER_DISABLE_IT`**Description:**

- Disable the RTC Tamper interrupt.

**Parameters:**

- `__HANDLE__`: specifies the RTC handle.
- `__INTERRUPT__`: specifies the RTC Tamper interrupt sources to be disabled. This parameter can be any combination of the following values:
  - `RTC_IT_TAMP`: Tamper interrupt

**Return value:**

- None

### `_HAL_RTC_TAMPER_GET_IT`

#### **Description:**

- Check whether the specified RTC Tamper interrupt has occurred or not.

#### **Parameters:**

- `_HANDLE_`: specifies the RTC handle.
- `_INTERRUPT_`: specifies the RTC Tamper interrupt to check. This parameter can be:
  - `RTC_IT_TAMP1`: Tamper1 interrupt
  - `RTC_IT_TAMP2`: Tamper2 interrupt
  - `RTC_IT_TAMP3`: Tamper3 interrupt (\*)

#### **Return value:**

- None

#### **Notes:**

- (\*) `RTC_IT_TAMP3` not present on all the devices

### `_HAL_RTC_TAMPER_GET_IT_SOURCE`

#### **Description:**

- Check whether the specified RTC Tamper interrupt has been enabled or not.

#### **Parameters:**

- `_HANDLE_`: specifies the RTC handle.
- `_INTERRUPT_`: specifies the RTC Tamper interrupt source to check. This parameter can be:
  - `RTC_IT_TAMP`: Tamper interrupt

#### **Return value:**

- None

### `_HAL_RTC_TAMPER_GET_FLAG`

#### **Description:**

- Get the selected RTC Tamper's flag status.

#### **Parameters:**

- `_HANDLE_`: specifies the RTC handle.
- `_FLAG_`: specifies the RTC Tamper Flag is pending or not. This parameter can be:
  - `RTC_FLAG_TAMP1F`
  - `RTC_FLAG_TAMP2F`

- RTC\_FLAG\_TAMP3F (\*)

**Return value:**

- None

**Notes:**

- (\*) RTC\_FLAG\_TAMP3F not present on all the devices

[\\_\\_HAL\\_RTC\\_TAMPER\\_CLEAR\\_FLAG](#)**Description:**

- Clear the RTC Tamper's pending flags.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the RTC handle.
- [\\_\\_FLAG\\_\\_](#): specifies the RTC Tamper Flag to clear. This parameter can be:
  - RTC\_FLAG\_TAMP1F
  - RTC\_FLAG\_TAMP2F
  - RTC\_FLAG\_TAMP3F (\*)

**Return value:**

- None

**Notes:**

- (\*) RTC\_FLAG\_TAMP3F not present on all the devices

***RTC Extended Tamper Filter Definition***

|                                                 |                                                                      |
|-------------------------------------------------|----------------------------------------------------------------------|
| <u><a href="#">RTC_TAMPERFILTER_DISABLE</a></u> | Tamper filter is disabled                                            |
| <u><a href="#">RTC_TAMPERFILTER_2SAMPLE</a></u> | Tamper is activated after 2 consecutive samples at the active level  |
| <u><a href="#">RTC_TAMPERFILTER_4SAMPLE</a></u> | Tamper is activated after 4 consecutive samples at the active level  |
| <u><a href="#">RTC_TAMPERFILTER_8SAMPLE</a></u> | Tamper is activated after 8 consecutive samples at the active level. |

***RTC Extended Tamper Pins Definition***

|                                     |
|-------------------------------------|
| <u><a href="#">RTC_TAMPER_1</a></u> |
| <u><a href="#">RTC_TAMPER_2</a></u> |
| <u><a href="#">RTC_TAMPER_3</a></u> |

***RTC Extended Tamper Pin Precharge Duration Definition***

|                                                            |                                                                    |
|------------------------------------------------------------|--------------------------------------------------------------------|
| <u><a href="#">RTC_TAMPERPRECHARGEDURATION_1RTCCLK</a></u> | Tamper pins are pre-charged before sampling during 1 RTCCLK cycle  |
| <u><a href="#">RTC_TAMPERPRECHARGEDURATION_2RTCCLK</a></u> | Tamper pins are pre-charged before sampling during 2 RTCCLK cycles |
| <u><a href="#">RTC_TAMPERPRECHARGEDURATION_4RTCCLK</a></u> | Tamper pins are pre-charged before sampling during 4 RTCCLK        |

|                                                                   |                                                                          |
|-------------------------------------------------------------------|--------------------------------------------------------------------------|
|                                                                   | cycles                                                                   |
| RTC_TAMPERPRECHARGEDURATION_8RTCCLK                               | Tamper pins are pre-charged before sampling during 8 RTCCLK cycles       |
| <b><i>RTC Extended Tamper Pull UP Definition</i></b>              |                                                                          |
| RTC_TAMPER_PULLUP_ENABLE                                          | Tamper pins are pre-charged before sampling                              |
| RTC_TAMPER_PULLUP_DISABLE                                         | Tamper pins are not pre-charged before sampling                          |
| <b><i>RTC Extended Tamper Sampling Frequencies Definition</i></b> |                                                                          |
| RTC_TAMPERSAMPLINGFREQ_RTCCLK_DIV32768                            | Each of the tamper inputs are sampled with a frequency = RTCCLK / 32768U |
| RTC_TAMPERSAMPLINGFREQ_RTCCLK_DIV16384                            | Each of the tamper inputs are sampled with a frequency = RTCCLK / 16384U |
| RTC_TAMPERSAMPLINGFREQ_RTCCLK_DIV8192                             | Each of the tamper inputs are sampled with a frequency = RTCCLK / 8192   |
| RTC_TAMPERSAMPLINGFREQ_RTCCLK_DIV4096                             | Each of the tamper inputs are sampled with a frequency = RTCCLK / 4096   |
| RTC_TAMPERSAMPLINGFREQ_RTCCLK_DIV2048                             | Each of the tamper inputs are sampled with a frequency = RTCCLK / 2048   |
| RTC_TAMPERSAMPLINGFREQ_RTCCLK_DIV1024                             | Each of the tamper inputs are sampled with a frequency = RTCCLK / 1024   |
| RTC_TAMPERSAMPLINGFREQ_RTCCLK_DIV512                              | Each of the tamper inputs are sampled with a frequency = RTCCLK / 512    |
| RTC_TAMPERSAMPLINGFREQ_RTCCLK_DIV256                              | Each of the tamper inputs are sampled with a frequency = RTCCLK / 256    |

***EXTI RTC Extended Tamper Timestamp EXTI***

| <u>_HAL_RTC_TAMPER_TIMESTAMP_EXTI_ENABLE_IT</u>         | <b>Description:</b>                                                                                                        |
|---------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
|                                                         | <ul style="list-style-type: none"> <li>• Enable interrupt on the RTC Tamper and Timestamp associated Exti line.</li> </ul> |
| <b>Return value:</b>                                    |                                                                                                                            |
|                                                         | <ul style="list-style-type: none"> <li>• None</li> </ul>                                                                   |
| <b><u>_HAL_RTC_TAMPER_TIMESTAMP_EXTI_DISABLE_IT</u></b> |                                                                                                                            |
| <b>Description:</b>                                     |                                                                                                                            |
|                                                         | <ul style="list-style-type: none"> <li>• Disable interrupt on the RTC Tamper and Timestamp associated</li> </ul>           |

Exti line.

**Return value:**

- None

**Description:**

- Enable event on the RTC Tamper and Timestamp associated Exti line.

**Return value:**

- None.

**Description:**

- Disable event on the RTC Tamper and Timestamp associated Exti line.

**Return value:**

- None.

**Description:**

- Enable falling edge trigger on the RTC Tamper and Timestamp associated Exti line.

**Return value:**

- None.

**Description:**

- Disable falling edge trigger on the RTC Tamper and Timestamp associated Exti line.

**Return value:**

- None.

**Description:**

- Enable rising edge trigger on the RTC Tamper and Timestamp associated Exti line.

**Return value:**

- None.

**Description:**

- Disable rising edge trigger on the RTC Tamper and Timestamp

associated Exti line.

**Return value:**

- None.

**Description:**

- Enable rising & falling edge trigger on the RTC Tamper and Timestamp associated Exti line.

**Return value:**

- None.

**Description:**

- Disable rising & falling edge trigger on the RTC Tamper and Timestamp associated Exti line.

**Return value:**

- None.

**Description:**

- Check whether the RTC Tamper and Timestamp associated Exti line interrupt flag is set or not.

**Return value:**

- Line: Status.

**Description:**

- Clear the RTC Tamper and Timestamp associated Exti line flag.

**Return value:**

- None.

**Description:**

- Generate a Software interrupt on the RTC Tamper and Timestamp associated Exti line.

**Return value:**

- None.

`__HAL_RTC_TAMPER_TIMESTAMP_EXTI_ENABLE_RISING_FALLING_EDGE`

`__HAL_RTC_TAMPER_TIMESTAMP_EXTI_DISABLE_RISING_FALLING_EDGE`

`__HAL_RTC_TAMPER_TIMESTAMP_EXTI_GET_FLAG`

`__HAL_RTC_TAMPER_TIMESTAMP_EXTI_CLEAR_FLAG`

`__HAL_RTC_TAMPER_TIMESTAMP_EXTI_GENERATE_SWIT`

***RTC Extended Tamper TimeStampOnTamperDetection Definition***

|                                                    |                                           |
|----------------------------------------------------|-------------------------------------------|
| <code>RTC_TIMESTAMPONTAMPERDETECTION_ENABLE</code> | TimeStamp on Tamper Detection event saved |
|----------------------------------------------------|-------------------------------------------|

`RTC_TIMESTAMPONTAMPERDETECTION_DISABLE` TimeStamp on Tamper Detection event is not saved

***RTC Extended Tamper Trigger Definition***

`RTC_TAMPERTRIGGER_RISINGEDGE`  
`RTC_TAMPERTRIGGER_FALLINGEDGE`  
`RTC_TAMPERTRIGGER_LOWLEVEL`  
`RTC_TAMPERTRIGGER_HIGHLEVEL`

***RTC Extended Timestamp***

`_HAL_RTC_TIMESTAMP_ENABLE`

**Description:**

- Enable the RTC TimeStamp peripheral.

**Parameters:**

- `_HANDLE_`: specifies the RTC handle.

**Return value:**

- None

`_HAL_RTC_TIMESTAMP_DISABLE`

**Description:**

- Disable the RTC TimeStamp peripheral.

**Parameters:**

- `_HANDLE_`: specifies the RTC handle.

**Return value:**

- None

`_HAL_RTC_TIMESTAMP_ENABLE_IT`

**Description:**

- Enable the RTC TimeStamp interrupt.

**Parameters:**

- `_HANDLE_`: specifies the RTC handle.
- `_INTERRUPT_`: specifies the RTC TimeStamp interrupt source to be enabled. This parameter can be:
  - `RTC_IT_TS`: TimeStamp interrupt

**Return value:**

- None

`_HAL_RTC_TIMESTAMP_DISABLE_IT`

**Description:**

- Disable the RTC TimeStamp interrupt.

**Parameters:**

- \_\_HANDLE\_\_: specifies the RTC handle.
- \_\_INTERRUPT\_\_: specifies the RTC TimeStamp interrupt source to be disabled. This parameter can be:
  - RTC\_IT\_TS: TimeStamp interrupt

**Return value:**

- None

\_HAL\_RTC\_TIMESTAMP\_GET\_IT

**Description:**

- Check whether the specified RTC TimeStamp interrupt has occurred or not.

**Parameters:**

- \_\_HANDLE\_\_: specifies the RTC handle.
- \_\_INTERRUPT\_\_: specifies the RTC TimeStamp interrupt to check. This parameter can be:
  - RTC\_IT\_TS: TimeStamp interrupt

**Return value:**

- None

\_HAL\_RTC\_TIMESTAMP\_GET\_IT\_SOURCE

**Description:**

- Check whether the specified RTC Time Stamp interrupt has been enabled or not.

**Parameters:**

- \_\_HANDLE\_\_: specifies the RTC handle.
- \_\_INTERRUPT\_\_: specifies the RTC Time Stamp interrupt source to check. This parameter can be:
  - RTC\_IT\_TS: TimeStamp interrupt

**Return value:**

- None

\_HAL\_RTC\_TIMESTAMP\_GET\_FLAG

**Description:**

- Get the selected RTC TimeStamp's flag status.

**Parameters:**

- \_\_HANDLE\_\_: specifies the RTC handle.
- \_\_FLAG\_\_: specifies the RTC

TimeStamp Flag is pending or not.

This parameter can be:

- RTC\_FLAG\_TSF
- RTC\_FLAG\_TSOVF

**Return value:**

- None

`_HAL_RTC_TIMESTAMP_CLEAR_FLAG`

**Description:**

- Clear the RTC Time Stamp's pending flags.

**Parameters:**

- `_HANDLE_`: specifies the RTC handle.
- `_FLAG_`: specifies the RTC Alarm Flag to clear. This parameter can be:
  - RTC\_FLAG\_TSF

**Return value:**

- None

***RTC Extended TimeStamp Pin Selection***

`RTC_TIMESTAMPPIN_DEFAULT`

***RTC Extended Time Stamp Edges definition***

`RTC_TIMESTAMPEDGE_RISING`

`RTC_TIMESTAMPEDGE_FALLING`

***RTC Extended WakeUp Timer***

`_HAL_RTC_WAKEUPTIMER_ENABLE`

**Description:**

- Enable the RTC WakeUp Timer peripheral.

**Parameters:**

- `_HANDLE_`: specifies the RTC handle.

**Return value:**

- None

`_HAL_RTC_WAKEUPTIMER_DISABLE`

**Description:**

- Disable the RTC WakeUp Timer peripheral.

**Parameters:**

- `_HANDLE_`: specifies the RTC handle.

**Return value:**

- None

[\\_\\_HAL\\_RTC\\_WAKEUPTIMER\\_ENABLE\\_IT](#)**Description:**

- Enable the RTC WakeUpTimer interrupt.

**Parameters:**

- \_\_HANDLE\_\_: specifies the RTC handle.
- \_\_INTERRUPT\_\_: specifies the RTC WakeUpTimer interrupt sources to be enabled. This parameter can be:
  - RTC\_IT\_WUT: WakeUpTimer interrupt

**Return value:**

- None

[\\_\\_HAL\\_RTC\\_WAKEUPTIMER\\_DISABLE\\_IT](#)**Description:**

- Disable the RTC WakeUpTimer interrupt.

**Parameters:**

- \_\_HANDLE\_\_: specifies the RTC handle.
- \_\_INTERRUPT\_\_: specifies the RTC WakeUpTimer interrupt sources to be disabled. This parameter can be:
  - RTC\_IT\_WUT: WakeUpTimer interrupt

**Return value:**

- None

[\\_\\_HAL\\_RTC\\_WAKEUPTIMER\\_GET\\_IT](#)**Description:**

- Check whether the specified RTC WakeUpTimer interrupt has occurred or not.

**Parameters:**

- \_\_HANDLE\_\_: specifies the RTC handle.
- \_\_INTERRUPT\_\_: specifies the RTC WakeUpTimer interrupt to check. This parameter can be:
  - RTC\_IT\_WUT: WakeUpTimer interrupt

**Return value:**

- None

[\\_\\_HAL\\_RTC\\_WAKEUPTIMER\\_GET\\_IT\\_SOURCE](#)**Description:**

- Check whether the specified RTC Wake Up timer interrupt has been enabled or not.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the RTC handle.
- [\\_\\_INTERRUPT\\_\\_](#): specifies the RTC Wake Up timer interrupt sources to check. This parameter can be:
  - [RTC\\_IT\\_WUT](#): WakeUpTimer interrupt

**Return value:**

- None

[\\_\\_HAL\\_RTC\\_WAKEUPTIMER\\_GET\\_FLAG](#)**Description:**

- Get the selected RTC WakeUpTimer's flag status.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the RTC handle.
- [\\_\\_FLAG\\_\\_](#): specifies the RTC WakeUpTimer Flag is pending or not. This parameter can be:
  - [RTC\\_FLAG\\_WUTF](#)
  - [RTC\\_FLAG\\_WUTWF](#)

**Return value:**

- None

[\\_\\_HAL\\_RTC\\_WAKEUPTIMER\\_CLEAR\\_FLAG](#)**Description:**

- Clear the RTC Wake Up timer's pending flags.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the RTC handle.
- [\\_\\_FLAG\\_\\_](#): specifies the RTC WakeUpTimer Flag to clear. This parameter can be:
  - [RTC\\_FLAG\\_WUTF](#)

**Return value:**

- None

[\\_\\_HAL\\_RTC\\_WAKEUPTIMER\\_EXTI\\_ENABLE\\_IT](#)**Description:**

- Enable interrupt on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None

**Description:**

- Disable interrupt on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None

**Description:**

- Enable event on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None.

**Description:**

- Disable event on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None.

**Description:**

- Enable falling edge trigger on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None.

**Description:**

- Disable falling edge trigger on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None.

**Description:**

- Enable rising edge trigger on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None.

**Description:**

- Disable rising edge trigger on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None.

`__HAL_RTC_WAKEUPTIMER_EXTI_ENABLE  
_RISING_FALLING_EDGE`

**Description:**

- Enable rising & falling edge trigger on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None.

`__HAL_RTC_WAKEUPTIMER_EXTI_DISABLE  
_RISING_FALLING_EDGE`

**Description:**

- Disable rising & falling edge trigger on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None.

`__HAL_RTC_WAKEUPTIMER_EXTI_GET_  
FLAG`

**Description:**

- Check whether the RTC WakeUp Timer associated Exti line interrupt flag is set or not.

**Return value:**

- Line: Status.

**Description:**

- Clear the RTC WakeUp Timer associated Exti line flag.

**Return value:**

- None.

**Description:**

- Generate a Software interrupt on the RTC WakeUp Timer associated Exti line.

**Return value:**

- None.

***RTC Extended Wakeup Timer Definition***

`RTC_WAKEUPCLOCK_RTCCLK_DIV16`  
`RTC_WAKEUPCLOCK_RTCCLK_DIV8`  
`RTC_WAKEUPCLOCK_RTCCLK_DIV4`  
`RTC_WAKEUPCLOCK_RTCCLK_DIV2`  
`RTC_WAKEUPCLOCK_CK_SPRE_16BITS`  
`RTC_WAKEUPCLOCK_CK_SPRE_17BITS`

## 44 HAL SDADC Generic Driver

### 44.1 SDADC Firmware driver registers structures

#### 44.1.1 SDADC\_InitTypeDef

##### Data Fields

- *uint32\_t IdleLowPowerMode*
- *uint32\_t FastConversionMode*
- *uint32\_t SlowClockMode*
- *uint32\_t ReferenceVoltage*

##### Field Documentation

- ***uint32\_t SDADC\_InitTypeDef::IdleLowPowerMode***  
Specifies if SDADC can enter in power down or standby when idle. This parameter can be a value of [\*\*SDADC\\_Idle\\_Low\\_Power\\_Mode\*\*](#)
- ***uint32\_t SDADC\_InitTypeDef::FastConversionMode***  
Specifies if Fast conversion mode is enabled or not. This parameter can be a value of [\*\*SDADC\\_Fast\\_Conv\\_Mode\*\*](#)
- ***uint32\_t SDADC\_InitTypeDef::SlowClockMode***  
Specifies if slow clock mode is enabled or not. This parameter can be a value of [\*\*SDADC\\_Slow\\_Clock\\_Mode\*\*](#)
- ***uint32\_t SDADC\_InitTypeDef::ReferenceVoltage***  
Specifies the reference voltage. Note: This parameter is common to all SDADC instances. This parameter can be a value of [\*\*SDADC\\_Reference\\_Voltage\*\*](#)

#### 44.1.2 SDADC\_HandleTypeDefDef

##### Data Fields

- *SDADC\_TypeDef \* Instance*
- *SDADC\_InitTypeDef Init*
- *DMA\_HandleTypeDef \* hdma*
- *uint32\_t RegularContMode*
- *uint32\_t InjectedContMode*
- *uint32\_t InjectedChannelsNbr*
- *uint32\_t InjConvRemaining*
- *uint32\_t RegularTrigger*
- *uint32\_t InjectedTrigger*
- *uint32\_t ExtTriggerEdge*
- *uint32\_t RegularMultimode*
- *uint32\_t InjectedMultimode*
- *HAL\_SDADC\_StateTypeDef State*
- *uint32\_t ErrorCode*

##### Field Documentation

- ***SDADC\_TypeDef\* SDADC\_HandleTypeDefDef::Instance***  
SDADC registers base address
- ***SDADC\_InitTypeDef SDADC\_HandleTypeDefDef::Init***  
SDADC init parameters
- ***DMA\_HandleTypeDef\* SDADC\_HandleTypeDefDef::hdma***  
SDADC DMA Handle parameters

- ***uint32\_t SDADC\_HandleTypeDef::RegularContMode***  
Regular conversion continuous mode
- ***uint32\_t SDADC\_HandleTypeDef::InjectedContMode***  
Injected conversion continuous mode
- ***uint32\_t SDADC\_HandleTypeDef::InjectedChannelsNbr***  
Number of channels in injected sequence
- ***uint32\_t SDADC\_HandleTypeDef::InjConvRemaining***  
Injected conversion remaining
- ***uint32\_t SDADC\_HandleTypeDef::RegularTrigger***  
Current trigger used for regular conversion
- ***uint32\_t SDADC\_HandleTypeDef::InjectedTrigger***  
Current trigger used for injected conversion
- ***uint32\_t SDADC\_HandleTypeDef::ExtTriggerEdge***  
Rising, falling or both edges selected
- ***uint32\_t SDADC\_HandleTypeDef::RegularMultimode***  
current type of regular multimode
- ***uint32\_t SDADC\_HandleTypeDef::InjectedMultimode***  
Current type of injected multimode
- ***HAL\_SDADC\_StateTypeDef SDADC\_HandleTypeDef::State***  
SDADC state
- ***uint32\_t SDADC\_HandleTypeDef::ErrorCode***  
SDADC Error code

#### 44.1.3 SDADC\_ConfParamTypeDef

##### Data Fields

- ***uint32\_t InputMode***
- ***uint32\_t Gain***
- ***uint32\_t CommonMode***
- ***uint32\_t Offset***

##### Field Documentation

- ***uint32\_t SDADC\_ConfParamTypeDef::InputMode***  
Specifies the input mode (single ended, differential...) This parameter can be any value of [\*\*SDADC\\_InputMode\*\*](#)
- ***uint32\_t SDADC\_ConfParamTypeDef::Gain***  
Specifies the gain setting. This parameter can be any value of [\*\*SDADC\\_Gain\*\*](#)
- ***uint32\_t SDADC\_ConfParamTypeDef::CommonMode***  
Specifies the common mode setting (VSSA, VDDA, VDDA/2U). This parameter can be any value of [\*\*SDADC\\_CommonMode\*\*](#)
- ***uint32\_t SDADC\_ConfParamTypeDef::Offset***  
Specifies the 12-bit offset value. This parameter can be any value lower or equal to 0x00000FFFU

## 44.2 SDADC Firmware driver API description

### 44.2.1 SDADC specific features

1. 16-bit sigma delta architecture.
2. Self calibration.
3. Interrupt generation at the end of calibration, regular/injected conversion and in case of overrun events.
4. Single and continuous conversion modes.
5. External trigger option with configurable polarity for injected conversion.

6. Multi mode (synchronized another SDADC with SDADC1).
7. DMA request generation during regular or injected channel conversion.

#### 44.2.2 How to use this driver

##### Initialization

1. As prerequisite, fill in the HAL\_SDADC\_MspInit() :
  - Enable SDADCx clock interface with \_\_SDADCx\_CLK\_ENABLE().
  - Configure SDADCx clock divider with HAL\_RCCEx\_PeriphCLKConfig.
  - Enable power on SDADC with HAL\_PWREx\_EnableSDADC().
  - Enable the clocks for the SDADC GPIOs with \_\_HAL\_RCC\_GPIOx\_CLK\_ENABLE().
  - Configure these SDADC pins in analog mode using HAL\_GPIO\_Init().
  - If interrupt mode is used, enable and configure SDADC global interrupt with HAL\_NVIC\_SetPriority() and HAL\_NVIC\_EnableIRQ().
  - If DMA mode is used, configure DMA with HAL\_DMA\_Init and link it with SDADC handle using \_\_HAL\_LINKDMA.
2. Configure the SDADC low power mode, fast conversion mode, slow clock mode and SDADC1 reference voltage using the HAL\_ADC\_Init() function. Note: Common reference voltage. is common to all SDADC instances.
3. Prepare channel configurations (input mode, common mode, gain and offset) using HAL\_SDADC\_PrepChannelConfig and associate channel with one configuration using HAL\_SDADC\_AssociateChannelConfig.

##### Calibration

1. Start calibration using HAL\_SDADC\_StartCalibration or HAL\_SDADC\_CalibrationStart\_IT.
2. In polling mode, use HAL\_SDADC\_PollForCalibEvent to detect the end of calibration.
3. In interrupt mode, HAL\_SDADC\_CalibrationCpltCallback will be called at the end of calibration.

##### Regular channel conversion

1. Select trigger for regular conversion using HAL\_SDADC\_SelectRegularTrigger.
2. Select regular channel and enable/disable continuous mode using HAL\_SDADC\_ConfigChannel.
3. Start regular conversion using HAL\_SDADC\_Start, HAL\_SDADC\_Start\_IT or HAL\_SDADC\_Start\_DMA.
4. In polling mode, use HAL\_SDADC\_PollForConversion to detect the end of regular conversion.
5. In interrupt mode, HAL\_SDADC\_ConvCpltCallback will be called at the end of regular conversion.
6. Get value of regular conversion using HAL\_SDADC\_GetValue.
7. In DMA mode, HAL\_SDADC\_ConvHalfCpltCallback and HAL\_SDADC\_ConvCpltCallback will be called respectively at the half transfer and at the transfer complete.
8. Stop regular conversion using HAL\_SDADC\_Stop, HAL\_SDADC\_Stop\_IT or HAL\_SDADC\_Stop\_DMA.

##### Injected channels conversion

1. Enable/disable delay on injected conversion using HAL\_SDADC\_SelectInjectedDelay.

2. If external trigger is used for injected conversion, configure this trigger using HAL\_SDADC\_SelectInjectedExtTrigger.
3. Select trigger for injected conversion using HAL\_SDADC\_SelectInjectedTrigger.
4. Select injected channels and enable/disable continuous mode using HAL\_SDADC\_InjectedConfigChannel.
5. Start injected conversion using HAL\_SDADC\_InjectedStart, HAL\_SDADC\_InjectedStart\_IT or HAL\_SDADC\_InjectedStart\_DMA.
6. In polling mode, use HAL\_SDADC\_PollForInjectedConversion to detect the end of injected conversion.
7. In interrupt mode, HAL\_SDADC\_InjectedConvCpltCallback will be called at the end of injected conversion.
8. Get value of injected conversion and corresponding channel using HAL\_SDADC\_InjectedGetValue.
9. In DMA mode, HAL\_SDADC\_InjectedConvHalfCpltCallback and HAL\_SDADC\_InjectedConvCpltCallback will be called respectively at the half transfer and at the transfer complete.
10. Stop injected conversion using HAL\_SDADC\_InjectedStop, HAL\_SDADC\_InjectedStop\_IT or HAL\_SDADC\_InjectedStop\_DMA.

### Multi mode regular channels conversions

1. Select type of multimode (SDADC1/SDADC2 or SDADC1/SDADC3) using HAL\_SDADC\_MultiModeConfigChannel.
2. Select software trigger for SDADC1 and synchronized trigger for SDADC2 (or SDADC3) using HAL\_SDADC\_SelectRegularTrigger.
3. Select regular channel for SDADC1 and SDADC2 (or SDADC3) using HAL\_SDADC\_ConfigChannel.
4. Start regular conversion for SDADC2 (or SDADC3) with HAL\_SDADC\_Start.
5. Start regular conversion for SDADC1 using HAL\_SDADC\_Start, HAL\_SDADC\_Start\_IT or HAL\_SDADC\_MultiModeStart\_DMA.
6. In polling mode, use HAL\_SDADC\_PollForConversion to detect the end of regular conversion for SDADC1.
7. In interrupt mode, HAL\_SDADC\_ConvCpltCallback will be called at the end of regular conversion for SDADC1.
8. Get value of regular conversions using HAL\_SDADC\_MultiModeGetValue.
9. In DMA mode, HAL\_SDADC\_ConvHalfCpltCallback and HAL\_SDADC\_ConvCpltCallback will be called respectively at the half transfer and at the transfer complete for SDADC1.
10. Stop regular conversion using HAL\_SDADC\_Stop, HAL\_SDADC\_Stop\_IT or HAL\_SDADC\_MultiModeStop\_DMA for SDADC1.
11. Stop regular conversion using HAL\_SDADC\_Stop for SDADC2 (or SDADC3).

### Multi mode injected channels conversions

1. Select type of multimode (SDADC1/SDADC2 or SDADC1/SDADC3) using HAL\_SDADC\_InjectedMultiModeConfigChannel.
2. Select software or external trigger for SDADC1 and synchronized trigger for SDADC2 (or SDADC3) using HAL\_SDADC\_SelectInjectedTrigger.
3. Select injected channels for SDADC1 and SDADC2 (or SDADC3) using HAL\_SDADC\_InjectedConfigChannel.
4. Start injected conversion for SDADC2 (or SDADC3) with HAL\_SDADC\_InjectedStart.
5. Start injected conversion for SDADC1 using HAL\_SDADC\_InjectedStart, HAL\_SDADC\_InjectedStart\_IT or HAL\_SDADC\_InjectedMultiModeStart\_DMA.
6. In polling mode, use HAL\_SDADC\_InjectedPollForConversion to detect the end of injected conversion for SDADC1.

7. In interrupt mode, HAL\_SDADC\_InjectedConvCpltCallback will be called at the end of injected conversion for SDADC1.
8. Get value of injected conversions using HAL\_SDADC\_InjectedMultiModeGetValue.
9. In DMA mode, HAL\_SDADC\_InjectedConvHalfCpltCallback and HAL\_SDADC\_InjectedConvCpltCallback will be called respectively at the half transfer and at the transfer complete for SDADC1.
10. Stop injected conversion using HAL\_SDADC\_InjectedStop, HAL\_SDADC\_InjectedStop\_IT or HAL\_SDADC\_InjectedMultiModeStop\_DMA for SDADC1.
11. Stop injected conversion using HAL\_SDADC\_InjectedStop for SDADC2 (or SDADC3).

#### 44.2.3 Initialization and de-initialization functions

This section provides functions allowing to:

- Initialize the SDADC.
- De-initialize the SDADC.

This section contains the following APIs:

- [\*HAL\\_SDADC\\_Init\(\)\*](#)
- [\*HAL\\_SDADC\\_DelInit\(\)\*](#)
- [\*HAL\\_SDADC\\_MspInit\(\)\*](#)
- [\*HAL\\_SDADC\\_MspDelInit\(\)\*](#)

#### 44.2.4 Peripheral control functions

This section provides functions allowing to:

- Program one of the three different configurations for channels.
- Associate channel to one of configurations.
- Select regular and injected channels.
- Enable/disable continuous mode for regular and injected conversions.
- Select regular and injected triggers.
- Select and configure injected external trigger.
- Enable/disable delay addition for injected conversions.
- Configure multimode.

This section contains the following APIs:

- [\*HAL\\_SDADC\\_PrepChannelConfig\(\)\*](#)
- [\*HAL\\_SDADC\\_AssociateChannelConfig\(\)\*](#)
- [\*HAL\\_SDADC\\_ConfigChannel\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedConfigChannel\(\)\*](#)
- [\*HAL\\_SDADC\\_SelectRegularTrigger\(\)\*](#)
- [\*HAL\\_SDADC\\_SelectInjectedTrigger\(\)\*](#)
- [\*HAL\\_SDADC\\_SelectInjectedExtTrigger\(\)\*](#)
- [\*HAL\\_SDADC\\_SelectInjectedDelay\(\)\*](#)
- [\*HAL\\_SDADC\\_MultiModeConfigChannel\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedMultiModeConfigChannel\(\)\*](#)

#### 44.2.5 IO operation functions

This section provides functions allowing to:

- Start calibration.
- Poll for the end of calibration.
- Start calibration and enable interrupt.

- Start conversion of regular/injected channel.
- Poll for the end of regular/injected conversion.
- Stop conversion of regular/injected channel.
- Start conversion of regular/injected channel and enable interrupt.
- Stop conversion of regular/injected channel and disable interrupt.
- Start conversion of regular/injected channel and enable DMA transfer.
- Stop conversion of regular/injected channel and disable DMA transfer.
- Start multimode and enable DMA transfer for regular/injected conversion.
- Stop multimode and disable DMA transfer for regular/injected conversion..
- Get result of regular channel conversion.
- Get result of injected channel conversion.
- Get result of multimode conversion.
- Handle SDADC interrupt request.
- Callbacks for calibration and regular/injected conversions.

This section contains the following APIs:

- [\*HAL\\_SDADC\\_CalibrationStart\(\)\*](#)
- [\*HAL\\_SDADC\\_PollForCalibEvent\(\)\*](#)
- [\*HAL\\_SDADC\\_CalibrationStart\\_IT\(\)\*](#)
- [\*HAL\\_SDADC\\_Start\(\)\*](#)
- [\*HAL\\_SDADC\\_PollForConversion\(\)\*](#)
- [\*HAL\\_SDADC\\_Stop\(\)\*](#)
- [\*HAL\\_SDADC\\_Start\\_IT\(\)\*](#)
- [\*HAL\\_SDADC\\_Stop\\_IT\(\)\*](#)
- [\*HAL\\_SDADC\\_Start\\_DMA\(\)\*](#)
- [\*HAL\\_SDADC\\_Stop\\_DMA\(\)\*](#)
- [\*HAL\\_SDADC\\_GetValue\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedStart\(\)\*](#)
- [\*HAL\\_SDADC\\_PollForInjectedConversion\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedStop\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedStart\\_IT\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedStop\\_IT\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedStart\\_DMA\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedStop\\_DMA\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedGetValue\(\)\*](#)
- [\*HAL\\_SDADC\\_MultiModeStart\\_DMA\(\)\*](#)
- [\*HAL\\_SDADC\\_MultiModeStop\\_DMA\(\)\*](#)
- [\*HAL\\_SDADC\\_MultiModeGetValue\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedMultiModeStart\\_DMA\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedMultiModeStop\\_DMA\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedMultiModeGetValue\(\)\*](#)
- [\*HAL\\_SDADC\\_IRQHandler\(\)\*](#)
- [\*HAL\\_SDADC\\_CalibrationCpltCallback\(\)\*](#)
- [\*HAL\\_SDADC\\_ConvHalfCpltCallback\(\)\*](#)
- [\*HAL\\_SDADC\\_ConvCpltCallback\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedConvHalfCpltCallback\(\)\*](#)
- [\*HAL\\_SDADC\\_InjectedConvCpltCallback\(\)\*](#)
- [\*HAL\\_SDADC\\_ErrorCallback\(\)\*](#)

#### 44.2.6 ADC Peripheral State functions

This subsection provides functions allowing to

- Get the SDADC state
- Get the SDADC Error

This section contains the following APIs:

- [\*HAL\\_SDADC\\_GetState\(\)\*](#)
- [\*HAL\\_SDADC\\_GetError\(\)\*](#)

#### 44.2.7 Detailed description of functions

##### **HAL\_SDADC\_Init**

|                      |                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_Init<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                                                         |
| Function description | Initializes the SDADC according to the specified parameters in the SDADC_InitTypeDef structure.                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdac</b>: SDADC handle.</li> </ul>                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL</b>: status.</li> </ul>                                                                            |
| Notes                | <ul style="list-style-type: none"> <li>• If multiple SDADC are used, please configure first SDADC1 to set the common reference voltage.</li> </ul> |

##### **HAL\_SDADC\_DelInit**

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_DelInit<br/>(SDADC_HandleTypeDef * hsdadc)</b>   |
| Function description | De-initializes the SDADC.                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdac</b>: SDADC handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL</b>: status.</li> </ul>         |

##### **HAL\_SDADC\_MspInit**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SDADC_MspInit (SDADC_HandleTypeDef * hsdadc)</b>                   |
| Function description | Initializes the SDADC MSP.                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdac</b>: SDADC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                |

##### **HAL\_SDADC\_MspDeInit**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SDADC_MspDeInit (SDADC_HandleTypeDef * hsdadc)</b>                 |
| Function description | De-initializes the SDADC MSP.                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdac</b>: SDADC handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                |

**HAL\_SDADC\_PrepChannelConfig**

|                      |                                                                                                                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_PrepChannelConfig<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t ConflIndex,<br/>SDADC_ConfParamTypeDef * ConfParamStruct)</b>                                                                                                                                  |
| Function description | This function allows the user to set parameters for a configuration.                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>ConflIndex:</b> Index of configuration to modify. This parameter can be a value of SDADC Configuration Index.</li> <li>• <b>ConfParamStruct:</b> Parameters to apply for this configuration.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state (neither calibration nor regular or injected conversion ongoing)</li> </ul>                                                                                               |

**HAL\_SDADC\_AssociateChannelConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_AssociateChannelConfig<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t Channel, uint32_t ConflIndex)</b>                                                                                                                                                                                                                        |
| Function description | This function allows the user to associate a channel with one of the available configurations.                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>Channel:</b> Channel to associate with configuration. This parameter can be a value of SDADC Channel Selection.</li> <li>• <b>ConflIndex:</b> Index of configuration to associate with channel. This parameter can be a value of SDADC Configuration Index.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state (neither calibration nor regular or injected conversion ongoing)</li> </ul>                                                                                                                                                              |

**HAL\_SDADC\_ConfigChannel**

|                      |                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_ConfigChannel<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t Channel, uint32_t ContinuousMode)</b>                                                                                                                                                                                                                         |
| Function description | This function allows to select channel for regular conversion and to enable/disable continuous mode for regular conversion.                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>Channel:</b> Channel for regular conversion. This parameter can be a value of SDADC Channel Selection.</li> <li>• <b>ContinuousMode:</b> Enable/disable continuous mode for regular conversion. This parameter can be a value of SDADC Continuous Mode.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                 |

**HAL\_SDADC\_InjectedConfigChannel**

|                      |                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_InjectedConfigChannel<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t Channel, uint32_t ContinuousMode)</b>                                                                                                                                                                                                                                 |
| Function description | This function allows to select channels for injected conversion and to enable/disable continuous mode for injected conversion.                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>Channel:</b> Channels for injected conversion. This parameter can be a values combination of SDADC Channel Selection.</li> <li>• <b>ContinuousMode:</b> Enable/disable continuous mode for injected conversion. This parameter can be a value of SDADC Continuous Mode.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                 |

**HAL\_SDADC\_SelectInjectedExtTrigger**

|                      |                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_SelectInjectedExtTrigger<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t InjectedExtTrigger, uint32_t ExtTriggerEdge)</b>                                                                                                                                                                                                                   |
| Function description | This function allows to select and configure injected external trigger.                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>InjectedExtTrigger:</b> External trigger for injected conversions. This parameter can be a value of SDADC Injected External Trigger.</li> <li>• <b>ExtTriggerEdge:</b> Edge of external injected trigger. This parameter can be a value of SDADC External Trigger Edge.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                 |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state (neither calibration nor regular or injected conversion ongoing)</li> </ul>                                                                                                                                                                          |

**HAL\_SDADC\_SelectInjectedDelay**

|                      |                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_SelectInjectedDelay<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t InjectedDelay)</b>                                                                                                                  |
| Function description | This function allows to enable/disable delay addition for injected conversions.                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>InjectedDelay:</b> Enable/disable delay for injected conversions. This parameter can be a value of SDADC Injected Conversion Delay.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state (neither calibration nor regular or injected conversion ongoing)</li> </ul>                                      |

**HAL\_SDADC\_SelectRegularTrigger**

|                      |                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_SelectRegularTrigger<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t Trigger)</b>                                                                                                                                                                                                                                                                                 |
| Function description | This function allows to select trigger for regular conversions.                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>Trigger:</b> Trigger for regular conversions. This parameter can be one of the following value : <ul style="list-style-type: none"> <li>– SDADC_SOFTWARE_TRIGGER : Software trigger.</li> <li>– SDADC_SYNCHRONOUS_TRIGGER : Synchronous with SDADC1 (only for SDADC2 and SDADC3).</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• This function should not be called if regular conversion is ongoing.</li> </ul>                                                                                                                                                                                                                                                                     |

**HAL\_SDADC\_SelectInjectedTrigger**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_SelectInjectedTrigger<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t Trigger)</b>                                                                                                                                                                                                                                                                                                                                       |
| Function description | This function allows to select trigger for injected conversions.                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>Trigger:</b> Trigger for injected conversions. This parameter can be one of the following value : <ul style="list-style-type: none"> <li>– SDADC_SOFTWARE_TRIGGER : Software trigger.</li> <li>– SDADC_SYNCHRONOUS_TRIGGER : Synchronous with SDADC1 (only for SDADC2 and SDADC3).</li> <li>– SDADC_EXTERNAL_TRIGGER : External trigger.</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                              |
| Notes                | <ul style="list-style-type: none"> <li>• This function should not be called if injected conversion is ongoing.</li> </ul>                                                                                                                                                                                                                                                                                                                           |

**HAL\_SDADC\_MultiModeConfigChannel**

|                      |                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_MultiModeConfigChannel<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t MultimodeType)</b>                                                                                                |
| Function description | This function allows to configure multimode for regular conversions.                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>MultimodeType:</b> Type of multimode for regular conversions. This parameter can be a value of SDADC Multimode Type.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                              |
| Notes                | <ul style="list-style-type: none"> <li>• This function should not be called if regular conversion is ongoing and should be called only for SDADC1.</li> </ul>                                                       |

**HAL\_SDADC\_InjectedMultiModeConfigChannel**

|                      |                                                                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_InjectedMultiModeConfigChannel(</b><br><b>(SDADC_HandleTypeDef * hsdadc, uint32_t MultimodeType)</b>                                                                                  |
| Function description | This function allows to configure multimode for injected conversions.                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>MultimodeType:</b> Type of multimode for injected conversions. This parameter can be a value of SDADC Multimode Type.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>• This function should not be called if injected conversion is ongoing and should be called only for SDADC1.</li> </ul>                                                       |

**HAL\_SDADC\_CalibrationStart**

|                      |                                                                                                                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_CalibrationStart(</b><br><b>(SDADC_HandleTypeDef * hsdadc, uint32_t CalibrationSequence)</b>                                                                                |
| Function description | This function allows to start calibration in polling mode.                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>CalibrationSequence:</b> Calibration sequence. This parameter can be a value of SDADC Calibration Sequence.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state (neither calibration nor regular or injected conversion ongoing).</li> </ul>             |

**HAL\_SDADC\_CalibrationStart\_IT**

|                      |                                                                                                                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_CalibrationStart_IT(</b><br><b>(SDADC_HandleTypeDef * hsdadc, uint32_t CalibrationSequence)</b>                                                                             |
| Function description | This function allows to start calibration in interrupt mode.                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>CalibrationSequence:</b> Calibration sequence. This parameter can be a value of SDADC Calibration Sequence.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state (neither calibration nor regular or injected conversion ongoing).</li> </ul>             |

**HAL\_SDADC\_Start**

|                      |                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_Start<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                                                                        |
| Function description | This function allows to start regular conversion in polling mode.                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsadc:</b> SDADC handle.</li> </ul>                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state or if injected conversion is ongoing.</li> </ul> |

**HAL\_SDADC\_Start\_IT**

|                      |                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_Start_IT<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                                                                     |
| Function description | This function allows to start regular conversion in interrupt mode.                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsadc:</b> SDADC handle.</li> </ul>                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state or if injected conversion is ongoing.</li> </ul> |

**HAL\_SDADC\_Start\_DMA**

|                      |                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_Start_DMA<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t * pData, uint32_t Length)</b>                                                                                                                       |
| Function description | This function allows to start regular conversion in DMA mode.                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsadc:</b> SDADC handle.</li> <li>• <b>pData:</b> The destination buffer address.</li> <li>• <b>Length:</b> The length of data to be transferred from SDADC peripheral to memory.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                   |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state or if injected conversion is ongoing.</li> </ul>                                                                       |

**HAL\_SDADC\_Stop**

|                      |                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_Stop<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                                |
| Function description | This function allows to stop regular conversion in polling mode.                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsadc:</b> SDADC handle.</li> </ul>                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only if regular conversion is ongoing.</li> </ul> |

### HAL\_SDADC\_Stop\_IT

|                      |                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_Stop_IT<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                         |
| Function description | This function allows to stop regular conversion in interrupt mode.                                                    |
| Parameters           | <ul style="list-style-type: none"><li><b>hsdacd:</b> SDADC handle.</li></ul>                                          |
| Return values        | <ul style="list-style-type: none"><li><b>HAL:</b> status</li></ul>                                                    |
| Notes                | <ul style="list-style-type: none"><li>This function should be called only if regular conversion is ongoing.</li></ul> |

### HAL\_SDADC\_Stop\_DMA

|                      |                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_Stop_DMA<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                        |
| Function description | This function allows to stop regular conversion in DMA mode.                                                          |
| Parameters           | <ul style="list-style-type: none"><li><b>hsdacd:</b> SDADC handle.</li></ul>                                          |
| Return values        | <ul style="list-style-type: none"><li><b>HAL:</b> status</li></ul>                                                    |
| Notes                | <ul style="list-style-type: none"><li>This function should be called only if regular conversion is ongoing.</li></ul> |

### HAL\_SDADC\_InjectedStart

|                      |                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_InjectedStart<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                                                           |
| Function description | This function allows to start injected conversion in polling mode.                                                                                            |
| Parameters           | <ul style="list-style-type: none"><li><b>hsdacd:</b> SDADC handle.</li></ul>                                                                                  |
| Return values        | <ul style="list-style-type: none"><li><b>HAL:</b> status</li></ul>                                                                                            |
| Notes                | <ul style="list-style-type: none"><li>This function should be called only when SDADC instance is in idle state or if regular conversion is ongoing.</li></ul> |

### HAL\_SDADC\_InjectedStart\_IT

|                      |                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_InjectedStart_IT<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                                                        |
| Function description | This function allows to start injected conversion in interrupt mode.                                                                                          |
| Parameters           | <ul style="list-style-type: none"><li><b>hsdacd:</b> SDADC handle.</li></ul>                                                                                  |
| Return values        | <ul style="list-style-type: none"><li><b>HAL:</b> status</li></ul>                                                                                            |
| Notes                | <ul style="list-style-type: none"><li>This function should be called only when SDADC instance is in idle state or if regular conversion is ongoing.</li></ul> |

**HAL\_SDADC\_InjectedStart\_DMA**

|                      |                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_InjectedStart_DMA<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t * pData, uint32_t Length)</b>                                                                                                                |
| Function description | This function allows to start injected conversion in DMA mode.                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>pData:</b> The destination buffer address.</li> <li>• <b>Length:</b> The length of data to be transferred from SDADC peripheral to memory.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state or if regular conversion is ongoing.</li> </ul>                                                                         |

**HAL\_SDADC\_InjectedStop**

|                      |                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_InjectedStop<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                         |
| Function description | This function allows to stop injected conversion in polling mode.                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> </ul>                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only if injected conversion is ongoing.</li> </ul> |

**HAL\_SDADC\_InjectedStop\_IT**

|                      |                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_InjectedStop_IT<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                      |
| Function description | This function allows to stop injected conversion in interrupt mode.                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> </ul>                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only if injected conversion is ongoing.</li> </ul> |

**HAL\_SDADC\_InjectedStop\_DMA**

|                      |                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_InjectedStop_DMA<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                     |
| Function description | This function allows to stop injected conversion in DMA mode.                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> </ul>                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only if injected conversion is ongoing.</li> </ul> |

**HAL\_SDADC\_MultiModeStart\_DMA**

|                      |                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_MultiModeStart_DMA<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t * pData, uint32_t Length)</b>                                                                                                               |
| Function description | This function allows to start multimode regular conversions in DMA mode.                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>pData:</b> The destination buffer address.</li> <li>• <b>Length:</b> The length of data to be transferred from SDADC peripheral to memory.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state or if injected conversion is ongoing.</li> </ul>                                                                        |

**HAL\_SDADC\_MultiModeStop\_DMA**

|                      |                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_MultiModeStop_DMA<br/>(SDADC_HandleTypeDef * hsdadc)</b>                                   |
| Function description | This function allows to stop multimode regular conversions in DMA mode.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> </ul>                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only if regular conversion is ongoing.</li> </ul> |

**HAL\_SDADC\_InjectedMultiModeStart\_DMA**

|                      |                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef<br/>HAL_SDADC_InjectedMultiModeStart_DMA<br/>(SDADC_HandleTypeDef * hsdadc, uint32_t * pData, uint32_t Length)</b>                                                                                                   |
| Function description | This function allows to start multimode injected conversions in DMA mode.                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>pData:</b> The destination buffer address.</li> <li>• <b>Length:</b> The length of data to be transferred from SDADC peripheral to memory.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only when SDADC instance is in idle state or if regular conversion is ongoing.</li> </ul>                                                                         |

**HAL\_SDADC\_InjectedMultiModeStop\_DMA**

|                      |                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SDADC_InjectedMultiModeStop_DMA (SDADC_HandleTypeDef * hsdadc)</b>                                |
| Function description | This function allows to stop multimode injected conversions in DMA mode.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> </ul>                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be called only if injected conversion is ongoing.</li> </ul> |

**HAL\_SDADC\_GetValue**

|                      |                                                                                      |
|----------------------|--------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_SDADC_GetValue (SDADC_HandleTypeDef * hsdadc)</b>                    |
| Function description | This function allows to get regular conversion value.                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> </ul>     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Regular:</b> conversion value</li> </ul> |

**HAL\_SDADC\_InjectedGetValue**

|                      |                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_SDADC_InjectedGetValue (SDADC_HandleTypeDef * hsdadc, uint32_t * Channel)</b>                                                             |
| Function description | This function allows to get injected conversion value.                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> <li>• <b>Channel:</b> Corresponding channel of injected conversion.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Injected:</b> conversion value</li> </ul>                                                                     |

**HAL\_SDADC\_MultiModeGetValue**

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_SDADC_MultiModeGetValue (SDADC_HandleTypeDef * hsdadc)</b>                     |
| Function description | This function allows to get multimode regular conversion value.                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> </ul>               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Multimode:</b> regular conversion value</li> </ul> |

**HAL\_SDADC\_InjectedMultiModeGetValue**

|                      |                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_SDADC_InjectedMultiModeGetValue (SDADC_HandleTypeDef * hsdadc)</b>              |
| Function description | This function allows to get multimode injected conversion value.                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdadc:</b> SDADC handle.</li> </ul>                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Multimode:</b> injected conversion value</li> </ul> |

**HAL\_SDADC\_IRQHandler**

Function name      **void HAL\_SDADC\_IRQHandler (SDADC\_HandleTypeDef \* hsdadc)**

Function description      This function handles the SDADC interrupts.

Parameters      • **hsadc:** SDADC handle.

Return values      • **None**

**HAL\_SDADC\_PollForCalibEvent**

Function name      **HAL\_StatusTypeDef HAL\_SDADC\_PollForCalibEvent (SDADC\_HandleTypeDef \* hsdadc, uint32\_t Timeout)**

Function description      This function allows to poll for the end of calibration.

Parameters      • **hsadc:** SDADC handle.

                  • **Timeout:** Timeout value in milliseconds.

Return values      • **HAL:** status

Notes      • This function should be called only if calibration is ongoing.

**HAL\_SDADC\_PollForConversion**

Function name      **HAL\_StatusTypeDef HAL\_SDADC\_PollForConversion (SDADC\_HandleTypeDef \* hsdadc, uint32\_t Timeout)**

Function description      This function allows to poll for the end of regular conversion.

Parameters      • **hsadc:** SDADC handle.

                  • **Timeout:** Timeout value in milliseconds.

Return values      • **HAL:** status

Notes      • This function should be called only if regular conversion is ongoing.

**HAL\_SDADC\_PollForInjectedConversion**

Function name      **HAL\_StatusTypeDef HAL\_SDADC\_PollForInjectedConversion (SDADC\_HandleTypeDef \* hsdadc, uint32\_t Timeout)**

Function description      This function allows to poll for the end of injected conversion.

Parameters      • **hsadc:** SDADC handle.

                  • **Timeout:** Timeout value in milliseconds.

Return values      • **HAL:** status

Notes      • This function should be called only if injected conversion is ongoing.

**HAL\_SDADC\_CalibrationCpltCallback**

Function name      **void HAL\_SDADC\_CalibrationCpltCallback (SDADC\_HandleTypeDef \* hsdadc)**

Function description      Calibration complete callback.

---

|               |                                                                                |
|---------------|--------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"><li>• <b>hsdadc:</b> SDADC handle.</li></ul> |
| Return values | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                  |

**HAL\_SDADC\_ConvHalfCpltCallback**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SDADC_ConvHalfCpltCallback (SDADC_HandleTypeDef * hsdadc)</b>      |
| Function description | Half regular conversion complete callback.                                     |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hsdadc:</b> SDADC handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                  |

**HAL\_SDADC\_ConvCpltCallback**

|                      |                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SDADC_ConvCpltCallback (SDADC_HandleTypeDef * hsdadc)</b>                                                                                                            |
| Function description | Regular conversion complete callback.                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hsdadc:</b> SDADC handle.</li></ul>                                                                                                   |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                                    |
| Notes                | <ul style="list-style-type: none"><li>• In interrupt mode, user has to read conversion value in this function using HAL_SDADC_GetValue or HAL_SDADC_MultiModeGetValue.</li></ul> |

**HAL\_SDADC\_InjectedConvHalfCpltCallback**

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SDADC_InjectedConvHalfCpltCallback (SDADC_HandleTypeDef * hsdadc)</b> |
| Function description | Half injected conversion complete callback.                                       |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hsdadc:</b> SDADC handle.</li></ul>    |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                     |

**HAL\_SDADC\_InjectedConvCpltCallback**

|                      |                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SDADC_InjectedConvCpltCallback (SDADC_HandleTypeDef * hsdadc)</b>                                                                                                                    |
| Function description | Injected conversion complete callback.                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hsdadc:</b> SDADC handle.</li></ul>                                                                                                                   |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"><li>• In interrupt mode, user has to read conversion value in this function using HAL_SDADC_InjectedGetValue or HAL_SDADC_InjectedMultiModeGetValue.</li></ul> |

**HAL\_SDADC\_ErrorCallback**

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <code>void HAL_SDADC_ErrorCallback (SDADC_HandleTypeDef * hsdadc)</code>         |
| Function description | Error callback.                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdacd:</b> SDADC handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                  |

**HAL\_SDADC\_GetState**

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <code>HAL_SDADC_StateTypeDef HAL_SDADC_GetState (SDADC_HandleTypeDef * hsdadc)</code> |
| Function description | This function allows to get the current SDADC state.                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdacd:</b> SDADC handle.</li> </ul>      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>SDADC:</b> state.</li> </ul>              |

**HAL\_SDADC\_GetError**

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <code>uint32_t HAL_SDADC_GetError (SDADC_HandleTypeDef * hsdadc)</code>          |
| Function description | This function allows to get the current SDADC error code.                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsdacd:</b> SDADC handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>SDADC:</b> error code.</li> </ul>    |

## 44.3 SDADC Firmware driver defines

### 44.3.1 SDADC

***SDADC Calibration Sequence***

|                                      |                                                                                                                             |
|--------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| <code>SDADC_CALIBRATION_SEQ_1</code> | One calibration sequence to calculate offset of conf0 (OFFSET0[11:0])                                                       |
| <code>SDADC_CALIBRATION_SEQ_2</code> | Two calibration sequences to calculate offset of conf0 and conf1 (OFFSET0[11:0] and OFFSET1[11:0])                          |
| <code>SDADC_CALIBRATION_SEQ_3</code> | Three calibration sequences to calculate offset of conf0, conf1 and conf2 (OFFSET0[11:0], OFFSET1[11:0], and OFFSET2[11:0]) |

***SDADC Channel Selection***

|                              |
|------------------------------|
| <code>SDADC_CHANNEL_0</code> |
| <code>SDADC_CHANNEL_1</code> |
| <code>SDADC_CHANNEL_2</code> |
| <code>SDADC_CHANNEL_3</code> |
| <code>SDADC_CHANNEL_4</code> |
| <code>SDADC_CHANNEL_5</code> |
| <code>SDADC_CHANNEL_6</code> |

`SDADC_CHANNEL_7`

`SDADC_CHANNEL_8`

#### ***SDADC Common Mode***

`SDADC_COMMON_MODE_VSSA` Select SDADC VSSA as common mode

`SDADC_COMMON_MODE_VDDA_2` Select SDADC VDDA/2 as common mode

`SDADC_COMMON_MODE_VDDA` Select SDADC VDDA as common mode

#### ***SDADC Configuration Index***

`SDADC_CONF_INDEX_0` Configuration 0 Register selected

`SDADC_CONF_INDEX_1` Configuration 1 Register selected

`SDADC_CONF_INDEX_2` Configuration 2 Register selected

#### ***SDADC Continuous Mode***

`SDADC_CONTINUOUS_CONV_OFF` Conversion are not continuous

`SDADC_CONTINUOUS_CONV_ON` Conversion are continuous

#### ***SDADC Error Code***

`SDADC_ERROR_NONE` No error

`SDADC_ERROR_REGULAR_OVERRUN` Overrun occurs during regular conversion

`SDADC_ERROR_INJECTED_OVERRUN` Overrun occurs during injected conversion

`SDADC_ERROR_DMA` DMA error occurs

#### ***SDADC Exported Macros***

`_HAL_SDADC_ENABLE_IT`

##### **Description:**

- Enable the ADC end of conversion interrupt.

##### **Parameters:**

- `_HANDLE_`: ADC handle
- `_INTERRUPT_`: ADC Interrupt This parameter can be any combination of the following values:
  - `SDADC_IT_EOCAL`: End of calibration interrupt enable
  - `SDADC_IT_JEOC`: Injected end of conversion interrupt enable
  - `SDADC_IT_JOVR`: Injected data overrun interrupt enable
  - `SDADC_IT_REOC`: Regular end of conversion interrupt enable
  - `SDADC_IT_ROVR`: Regular data overrun interrupt enable

##### **Return value:**

- None

`_HAL_SDADC_DISABLE_IT`

##### **Description:**

- Disable the ADC end of conversion

interrupt.

**Parameters:**

- `__HANDLE__`: ADC handle
- `__INTERRUPT__`: ADC Interrupt This parameter can be any combination of the following values:
  - `SDADC_IT_EOCAL`: End of calibration interrupt enable
  - `SDADC_IT_JEOC`: Injected end of conversion interrupt enable
  - `SDADC_IT_JOVR`: Injected data overrun interrupt enable
  - `SDADC_IT_ROEC`: Regular end of conversion interrupt enable
  - `SDADC_IT_ROVR`: Regular data overrun interrupt enable

**Return value:**

- None

`__HAL_SDADC_GET_IT_SOURCE`

**Description:**

- Checks if the specified ADC interrupt source is enabled or disabled.

**Parameters:**

- `__HANDLE__`: ADC handle
- `__INTERRUPT__`: ADC interrupt source to check This parameter can be any combination of the following values:
  - `SDADC_IT_EOCAL`: End of calibration interrupt enable
  - `SDADC_IT_JEOC`: Injected end of conversion interrupt enable
  - `SDADC_IT_JOVR`: Injected data overrun interrupt enable
  - `SDADC_IT_ROEC`: Regular end of conversion interrupt enable
  - `SDADC_IT_ROVR`: Regular data overrun interrupt enable

**Return value:**

- State: of interruption (SET or RESET)

`__HAL_SDADC_GET_FLAG`

**Description:**

- Get the selected ADC's flag status.

**Parameters:**

- `__HANDLE__`: ADC handle
- `__FLAG__`: ADC flag This parameter can be any combination of the following values:
  - `SDADC_FLAG_EOCAL`: End of calibration flag

- SDADC\_FLAG\_JEOC: End of injected conversion flag
- SDADC\_FLAG\_JOVR: Injected conversion overrun flag
- SDADC\_FLAG\_REOC: End of regular conversion flag
- SDADC\_FLAG\_ROVR: Regular conversion overrun flag

**Return value:**

- None

`_HAL_SDADC_CLEAR_FLAG`**Description:**

- Clear the ADC's pending flags.

**Parameters:**

- `_HANDLE_`: ADC handle
- `_FLAG_`: ADC flag This parameter can be any combination of the following values:
  - SDADC\_FLAG\_EOCAL: End of calibration flag
  - SDADC\_FLAG\_JEOC: End of injected conversion flag
  - SDADC\_FLAG\_JOVR: Injected conversion overrun flag
  - SDADC\_FLAG\_REOC: End of regular conversion flag
  - SDADC\_FLAG\_ROVR: Regular conversion overrun flag

**Return value:**

- None

`_HAL_SDADC_RESET_HANDLE_STATE`**Description:**

- Reset SDADC handle state.

**Parameters:**

- `_HANDLE_`: SDADC handle.

**Return value:**

- None

***SDADC External Trigger Edge***

|                                          |                                   |
|------------------------------------------|-----------------------------------|
| <code>SDADC_EXT_TRIG_RISING_EDGE</code>  | External rising edge              |
| <code>SDADC_EXT_TRIG_FALLING_EDGE</code> | External falling edge             |
| <code>SDADC_EXT_TRIG_BOTH_EDGES</code>   | External rising and falling edges |

***SDADC Fast Conversion Mode***

|                                      |
|--------------------------------------|
| <code>SDADC_FAST_CONV_DISABLE</code> |
| <code>SDADC_FAST_CONV_ENABLE</code>  |

***SDADC flags definition***

|                  |                                  |
|------------------|----------------------------------|
| SDADC_FLAG_EOCAL | End of calibration flag          |
| SDADC_FLAG_JEOC  | End of injected conversion flag  |
| SDADC_FLAG_JOVR  | Injected conversion overrun flag |
| SDADC_FLAG_REOC  | End of regular conversion flag   |
| SDADC_FLAG_ROVR  | Regular conversion overrun flag  |

***SDADC Gain***

|                |                     |
|----------------|---------------------|
| SDADC_GAIN_1   | Gain equal to 1U    |
| SDADC_GAIN_2   | Gain equal to 2U    |
| SDADC_GAIN_4   | Gain equal to 4U    |
| SDADC_GAIN_8   | Gain equal to 8U    |
| SDADC_GAIN_16  | Gain equal to 16U   |
| SDADC_GAIN_32  | Gain equal to 32U   |
| SDADC_GAIN_1_2 | Gain equal to 1U/2U |

***SDADC Idle Low Power Mode***

|                          |  |
|--------------------------|--|
| SDADC_LOWPOWER_NONE      |  |
| SDADC_LOWPOWER_POWERDOWN |  |
| SDADC_LOWPOWER_STANDBY   |  |

***SDADC Injected Conversion Delay***

|                           |                                 |
|---------------------------|---------------------------------|
| SDADC_INJECTED_DELAY_NONE | No delay on injected conversion |
| SDADC_INJECTED_DELAY      | Delay on injected conversion    |

***SDADC Injected External Trigger***

|                          |                           |
|--------------------------|---------------------------|
| SDADC_EXT_TRIG_TIM13_CC1 | Trigger source for SDADC1 |
| SDADC_EXT_TRIG_TIM14_CC1 | Trigger source for SDADC1 |
| SDADC_EXT_TRIG_TIM16_CC1 | Trigger source for SDADC3 |
| SDADC_EXT_TRIG_TIM17_CC1 | Trigger source for SDADC2 |
| SDADC_EXT_TRIG_TIM12_CC1 | Trigger source for SDADC2 |
| SDADC_EXT_TRIG_TIM12_CC2 | Trigger source for SDADC3 |
| SDADC_EXT_TRIG_TIM15_CC2 | Trigger source for SDADC1 |
| SDADC_EXT_TRIG_TIM2_CC3  | Trigger source for SDADC2 |
| SDADC_EXT_TRIG_TIM2_CC4  | Trigger source for SDADC3 |
| SDADC_EXT_TRIG_TIM3_CC1  | Trigger source for SDADC1 |
| SDADC_EXT_TRIG_TIM3_CC2  | Trigger source for SDADC2 |
| SDADC_EXT_TRIG_TIM3_CC3  | Trigger source for SDADC3 |
| SDADC_EXT_TRIG_TIM4_CC1  | Trigger source for SDADC1 |
| SDADC_EXT_TRIG_TIM4_CC2  | Trigger source for SDADC2 |

|                                    |                                                                   |
|------------------------------------|-------------------------------------------------------------------|
| SDADC_EXT_TRIG_TIM4_CC3            | Trigger source for SDADC3                                         |
| SDADC_EXT_TRIG_TIM19_CC2           | Trigger source for SDADC1                                         |
| SDADC_EXT_TRIG_TIM19_CC3           | Trigger source for SDADC2                                         |
| SDADC_EXT_TRIG_TIM19_CC4           | Trigger source for SDADC3                                         |
| SDADC_EXT_TRIG EXTI11              | Trigger source for SDADC1, SDADC2 and SDADC3                      |
| SDADC_EXT_TRIG EXTI15              | Trigger source for SDADC1, SDADC2 and SDADC3                      |
| <b>SDADC Input Mode</b>            |                                                                   |
| SDADC_INPUT_MODE_DIFF              | Conversions are executed in differential mode                     |
| SDADC_INPUT_MODE_SE_OFFSET         | Conversions are executed in single ended offset mode              |
| SDADC_INPUT_MODE_SE_ZERO_REFERENCE | Conversions are executed in single ended zero-volt reference mode |
| <b>SDADC interrupts definition</b> |                                                                   |
| SDADC_IT_EOCAL                     | End of calibration interrupt enable                               |
| SDADC_IT_JEOC                      | Injected end of conversion interrupt enable                       |
| SDADC_IT_JOVR                      | Injected data overrun interrupt enable                            |
| SDADC_IT_REOC                      | Regular end of conversion interrupt enable                        |
| SDADC_IT_ROVR                      | Regular data overrun interrupt enable                             |
| <b>SDADC Multimode Type</b>        |                                                                   |
| SDADC_MULTIMODE_SDADC1_SDADC2      | Get conversion values for SDADC1 and SDADC2                       |
| SDADC_MULTIMODE_SDADC1_SDADC3      | Get conversion values for SDADC1 and SDADC3                       |
| <b>SDADC Reference Voltage</b>     |                                                                   |
| SDADC_VREF_EXT                     | The reference voltage is forced externally using VREF pin         |
| SDADC_VREF_VREFINT1                | The reference voltage is forced internally to 1.22V VREFINT       |
| SDADC_VREF_VREFINT2                | The reference voltage is forced internally to 1.8V VREFINT        |
| SDADC_VREF_VDDA                    | The reference voltage is forced internally to VDDA                |
| <b>SDADC Slow Clock Mode</b>       |                                                                   |
| SDADC_SLOW_CLOCK_DISABLE           |                                                                   |
| SDADC_SLOW_CLOCK_ENABLE            |                                                                   |
| <b>SDADC Trigger</b>               |                                                                   |
| SDADC_SOFTWARE_TRIGGER             | Software trigger                                                  |
| SDADC_SYNCHRONOUS_TRIGGER          | Synchronous with SDADC1 (only for SDADC2 and SDADC3)              |
| SDADC_EXTERNAL_TRIGGER             | External trigger                                                  |

## 45 HAL SMARTCARD Generic Driver

### 45.1 SMARTCARD Firmware driver registers structures

#### 45.1.1 SMARTCARD\_InitTypeDef

##### Data Fields

- *uint32\_t BaudRate*
- *uint32\_t WordLength*
- *uint32\_t StopBits*
- *uint16\_t Parity*
- *uint16\_t Mode*
- *uint16\_t CLKPolarity*
- *uint16\_t CLKPhase*
- *uint16\_t CLKLastBit*
- *uint16\_t OneBitSampling*
- *uint8\_t Prescaler*
- *uint8\_t GuardTime*
- *uint16\_t NACKEnable*
- *uint32\_t TimeOutEnable*
- *uint32\_t TimeOutValue*
- *uint8\_t BlockLength*
- *uint8\_t AutoRetryCount*

##### Field Documentation

- ***uint32\_t SMARTCARD\_InitTypeDef::BaudRate***  
Configures the SmartCard communication baud rate. The baud rate register is computed using the following formula: Baud Rate Register = ((PCLKx) / ((hsmcard->Init.BaudRate)))
- ***uint32\_t SMARTCARD\_InitTypeDef::WordLength***  
Specifies the number of data bits transmitted or received in a frame. This parameter ***SMARTCARD\_Word\_Length*** can only be set to 9 (8 data + 1 parity bits).
- ***uint32\_t SMARTCARD\_InitTypeDef::StopBits***  
Specifies the number of stop bits. This parameter can be a value of ***SMARTCARD\_Stop\_Bits***.
- ***uint16\_t SMARTCARD\_InitTypeDef::Parity***  
Specifies the parity mode. This parameter can be a value of ***SMARTCARD\_Parity***  
**Note:**The parity is enabled by default (PCE is forced to 1U). Since the WordLength is forced to 8 bits + parity, M is forced to 1 and the parity bit is the 9th bit.
- ***uint16\_t SMARTCARD\_InitTypeDef::Mode***  
Specifies whether the Receive or Transmit mode is enabled or disabled. This parameter can be a value of ***SMARTCARD\_Mode***
- ***uint16\_t SMARTCARD\_InitTypeDef::CLKPolarity***  
Specifies the steady state of the serial clock. This parameter can be a value of ***SMARTCARD\_Clock\_Polarity***
- ***uint16\_t SMARTCARD\_InitTypeDef::CLKPhase***  
Specifies the clock transition on which the bit capture is made. This parameter can be a value of ***SMARTCARD\_Clock\_Phase***
- ***uint16\_t SMARTCARD\_InitTypeDef::CLKLastBit***  
Specifies whether the clock pulse corresponding to the last transmitted data bit (MSB)

- has to be output on the SCLK pin in synchronous mode. This parameter can be a value of [\*\*SMARTCARD\\_Last\\_Bit\*\*](#).
- **`uint16_t SMARTCARD_InitTypeDef::OneBitSampling`**  
Specifies whether a single sample or three samples' majority vote is selected. Selecting the single sample method increases the receiver tolerance to clock deviations. This parameter can be a value of [\*\*SMARTCARD\\_OneBit\\_Sampling\*\*](#).
  - **`uint8_t SMARTCARD_InitTypeDef::Prescaler`**  
Specifies the SmartCard Prescaler.
  - **`uint8_t SMARTCARD_InitTypeDef::GuardTime`**  
Specifies the SmartCard Guard Time applied after stop bits.
  - **`uint16_t SMARTCARD_InitTypeDef::NACKEnable`**  
Specifies whether the SmartCard NACK transmission is enabled in case of parity error. This parameter can be a value of [\*\*SMARTCARD\\_NACK\\_Enable\*\*](#).
  - **`uint32_t SMARTCARD_InitTypeDef::TimeOutEnable`**  
Specifies whether the receiver timeout is enabled. This parameter can be a value of [\*\*SMARTCARD\\_Timeout\\_Enable\*\*](#).
  - **`uint32_t SMARTCARD_InitTypeDef::TimeOutValue`**  
Specifies the receiver time out value in number of baud blocks: it is used to implement the Character Wait Time (CWT) and Block Wait Time (BWT). It is coded over 24 bits.
  - **`uint8_t SMARTCARD_InitTypeDef::BlockLength`**  
Specifies the SmartCard Block Length in T=1 Reception mode. This parameter can be any value from 0x0 to 0xFFU.
  - **`uint8_t SMARTCARD_InitTypeDef::AutoRetryCount`**  
Specifies the SmartCard auto-retry count (number of retries in receive and transmit mode). When set to 0U, retransmission is disabled. Otherwise, its maximum value is 7 (before signalling an error).

#### 45.1.2 **SMARTCARD\_AdvFeatureInitTypeDef**

##### Data Fields

- **`uint32_t AdvFeatureInit`**
- **`uint32_t TxPinLevelInvert`**
- **`uint32_t RxPinLevelInvert`**
- **`uint32_t DataInvert`**
- **`uint32_t Swap`**
- **`uint32_t OverrunDisable`**
- **`uint32_t DMADisableonRxError`**
- **`uint32_t MSBFirst`**

##### Field Documentation

- **`uint32_t SMARTCARD_AdvFeatureInitTypeDef::AdvFeatureInit`**  
Specifies which advanced SMARTCARD features is initialized. Several advanced features may be initialized at the same time. This parameter can be a value of [\*\*SMARTCARD\\_Advanced\\_Features\\_Initialization\\_Type\*\*](#).
- **`uint32_t SMARTCARD_AdvFeatureInitTypeDef::TxPinLevelInvert`**  
Specifies whether the TX pin active level is inverted. This parameter can be a value of [\*\*SMARTCARD\\_Tx\\_Inv\*\*](#).
- **`uint32_t SMARTCARD_AdvFeatureInitTypeDef::RxPinLevelInvert`**  
Specifies whether the RX pin active level is inverted. This parameter can be a value of [\*\*SMARTCARD\\_Rx\\_Inv\*\*](#).
- **`uint32_t SMARTCARD_AdvFeatureInitTypeDef::DataInvert`**  
Specifies whether data are inverted (positive/direct logic vs negative/inverted logic). This parameter can be a value of [\*\*SMARTCARD\\_Data\\_Inv\*\*](#).

- **`uint32_t SMARTCARD_AdvFeatureInitTypeDef::Swap`**  
Specifies whether TX and RX pins are swapped. This parameter can be a value of `SMARTCARD_Rx_Tx_Swap`
- **`uint32_t SMARTCARD_AdvFeatureInitTypeDef::OverrunDisable`**  
Specifies whether the reception overrun detection is disabled. This parameter can be a value of `SMARTCARD_Overrun_Disable`
- **`uint32_t SMARTCARD_AdvFeatureInitTypeDef::DMADisableonRxError`**  
Specifies whether the DMA is disabled in case of reception error. This parameter can be a value of `SMARTCARD_DMA_Disable_on_Rx_Error`
- **`uint32_t SMARTCARD_AdvFeatureInitTypeDef::MSBFirst`**  
Specifies whether MSB is sent first on UART line. This parameter can be a value of `SMARTCARD_MSB_First`

### 45.1.3 SMARTCARD\_HandleTypeDef

#### Data Fields

- `USART_TypeDef * Instance`
- `SMARTCARD_InitTypeDef Init`
- `SMARTCARD_AdvFeatureInitTypeDef AdvancedInit`
- `uint8_t * pTxBuffPtr`
- `uint16_t TxXferSize`
- `_IO uint16_t TxXferCount`
- `uint8_t * pRxBuffPtr`
- `uint16_t RxXferSize`
- `_IO uint16_t RxXferCount`
- `DMA_HandleTypeDef * hdmatx`
- `DMA_HandleTypeDef * hdmarx`
- `HAL_LockTypeDef Lock`
- `_IO HAL_SMARTCARD_StateTypeDef gState`
- `_IO HAL_SMARTCARD_StateTypeDef RxState`
- `_IO uint32_t ErrorCode`

#### Field Documentation

- **`USART_TypeDef* SMARTCARD_HandleTypeDef::Instance`**  
USART registers base address
- **`SMARTCARD_InitTypeDef SMARTCARD_HandleTypeDef::Init`**  
SmartCard communication parameters
- **`SMARTCARD_AdvFeatureInitTypeDef SMARTCARD_HandleTypeDef::AdvancedInit`**  
SmartCard advanced features initialization parameters
- **`uint8_t* SMARTCARD_HandleTypeDef::pTxBuffPtr`**  
Pointer to SmartCard Tx transfer Buffer
- **`uint16_t SMARTCARD_HandleTypeDef::TxXferSize`**  
SmartCard Tx Transfer size
- **`_IO uint16_t SMARTCARD_HandleTypeDef::TxXferCount`**  
SmartCard Tx Transfer Counter
- **`uint8_t* SMARTCARD_HandleTypeDef::pRxBuffPtr`**  
Pointer to SmartCard Rx transfer Buffer
- **`uint16_t SMARTCARD_HandleTypeDef::RxXferSize`**  
SmartCard Rx Transfer size
- **`_IO uint16_t SMARTCARD_HandleTypeDef::RxXferCount`**  
SmartCard Rx Transfer Counter
- **`DMA_HandleTypeDef* SMARTCARD_HandleTypeDef::hdmatx`**  
SmartCard Tx DMA Handle parameters

- **DMA\_HandleTypeDef\* SMARTCARD\_HandleTypeDef::hdmarx**  
SmartCard Rx DMA Handle parameters
- **HAL\_LockTypeDef SMARTCARD\_HandleTypeDef::Lock**  
Locking object
- **\_IO HAL\_SMARTCARD\_StateTypeDef SMARTCARD\_HandleTypeDef::gState**  
SmartCard state information related to global Handle management and also related to Tx operations. This parameter can be a value of **HAL\_SMARTCARD\_StateTypeDef**
- **\_IO HAL\_SMARTCARD\_StateTypeDef SMARTCARD\_HandleTypeDef::RxState**  
SmartCard state information related to Rx operations. This parameter can be a value of **HAL\_SMARTCARD\_StateTypeDef**
- **\_IO uint32\_t SMARTCARD\_HandleTypeDef::ErrorCode**  
SmartCard Error code This parameter can be a value of **SMARTCARD\_Error**

## 45.2 SMARTCARD Firmware driver API description

### 45.2.1 How to use this driver

The SMARTCARD HAL driver can be used as follows:

1. Declare a SMARTCARD\_HandleTypeDef handle structure (eg. SMARTCARD\_HandleTypeDef hsmartcard).
2. Associate a USART to the SMARTCARD handle hsmartcard.
3. Initialize the SMARTCARD low level resources by implementing the HAL\_SMARTCARD\_MspInit() API:
  - Enable the USARTx interface clock.
  - USART pins configuration:
    - Enable the clock for the USART GPIOs.
    - Configure the USART pins (TX as alternate function pull-up, RX as alternate function Input).
  - NVIC configuration if you need to use interrupt process (HAL\_SMARTCARD\_Transmit\_IT() and HAL\_SMARTCARD\_Receive\_IT() APIs):
    - Configure the USARTx interrupt priority.
    - Enable the NVIC USART IRQ handle.
  - DMA Configuration if you need to use DMA process (HAL\_SMARTCARD\_Transmit\_DMA() and HAL\_SMARTCARD\_Receive\_DMA() APIs):
    - Declare a DMA handle structure for the Tx/Rx channel.
    - Enable the DMAx interface clock.
    - Configure the declared DMA handle structure with the required Tx/Rx parameters.
    - Configure the DMA Tx/Rx channel.
    - Associate the initialized DMA handle to the SMARTCARD DMA Tx/Rx handle.
    - Configure the priority and enable the NVIC for the transfer complete interrupt on the DMA Tx/Rx channel.
4. Program the Baud Rate, Parity, Mode(Receiver/Transmitter), clock enabling/disabling and accordingly, the clock parameters (parity, phase, last bit), prescaler value, guard time and NACK on transmission error enabling or disabling in the hsmartcard handle Init structure.
5. If required, program SMARTCARD advanced features (TX/RX pins swap, TimeOut, auto-retry counter,...) in the hsmartcard handle AdvancedInit structure.
6. Initialize the SMARTCARD registers by calling the HAL\_SMARTCARD\_Init() API:
  - This API configures also the low level Hardware (GPIO, CLOCK, CORTEX...etc) by calling the customized HAL\_SMARTCARD\_MspInit() API.



The specific SMARTCARD interrupts (Transmission complete interrupt, RXNE interrupt and Error Interrupts) will be managed using the macros `_HAL_SMARTCARD_ENABLE_IT()` and `_HAL_SMARTCARD_DISABLE_IT()` inside the transmit and receive process.

Three operation modes are available within this driver :

### **Polling mode IO operation**

- Send an amount of data in blocking mode using `HAL_SMARTCARD_Transmit()`
- Receive an amount of data in blocking mode using `HAL_SMARTCARD_Receive()`

### **Interrupt mode IO operation**

- Send an amount of data in non-blocking mode using `HAL_SMARTCARD_Transmit_IT()`
- At transmission end of transfer `HAL_SMARTCARD_TxCpltCallback()` is executed and user can add his own code by customization of function pointer `HAL_SMARTCARD_TxCpltCallback()`
- Receive an amount of data in non-blocking mode using `HAL_SMARTCARD_Receive_IT()`
- At reception end of transfer `HAL_SMARTCARD_RxCpltCallback()` is executed and user can add his own code by customization of function pointer `HAL_SMARTCARD_RxCpltCallback()`
- In case of transfer Error, `HAL_SMARTCARD_ErrorCallback()` function is executed and user can add his own code by customization of function pointer `HAL_SMARTCARD_ErrorCallback()`

### **DMA mode IO operation**

- Send an amount of data in non-blocking mode (DMA) using `HAL_SMARTCARD_Transmit_DMA()`
- At transmission end of transfer `HAL_SMARTCARD_TxCpltCallback()` is executed and user can add his own code by customization of function pointer `HAL_SMARTCARD_TxCpltCallback()`
- Receive an amount of data in non-blocking mode (DMA) using `HAL_SMARTCARD_Receive_DMA()`
- At reception end of transfer `HAL_SMARTCARD_RxCpltCallback()` is executed and user can add his own code by customization of function pointer `HAL_SMARTCARD_RxCpltCallback()`
- In case of transfer Error, `HAL_SMARTCARD_ErrorCallback()` function is executed and user can add his own code by customization of function pointer `HAL_SMARTCARD_ErrorCallback()`

### **SMARTCARD HAL driver macros list**

Below the list of most used macros in SMARTCARD HAL driver.

- `_HAL_SMARTCARD_GET_FLAG` : Check whether or not the specified SMARTCARD flag is set
- `_HAL_SMARTCARD_CLEAR_FLAG` : Clear the specified SMARTCARD pending flag
- `_HAL_SMARTCARD_ENABLE_IT`: Enable the specified SMARTCARD interrupt
- `_HAL_SMARTCARD_DISABLE_IT`: Disable the specified SMARTCARD interrupt

- `_HAL_SMARTCARD_GET_IT_SOURCE`: Check whether or not the specified SMARTCARD interrupt is enabled



You can refer to the SMARTCARD HAL driver header file for more useful macros

## 45.2.2 Initialization and Configuration functions

This subsection provides a set of functions allowing to initialize the USARTx associated to the SmartCard.

The Smartcard interface is designed to support asynchronous protocol Smartcards as defined in the ISO 7816-3 standard.

The USART can provide a clock to the smartcard through the SCLK output. In smartcard mode, SCLK is not associated to the communication but is simply derived from the internal peripheral input clock through a 5-bit prescaler.

- These parameters can be configured:
  - Baud Rate
  - Parity: should be enabled
  - Receiver/transmitter modes
  - Synchronous mode (and if enabled, phase, polarity and last bit parameters)
  - Prescaler value
  - Guard bit time
  - NACK enabling or disabling on transmission error
- The following advanced features can be configured as well:
  - TX and/or RX pin level inversion
  - data logical level inversion
  - RX and TX pins swap
  - RX overrun detection disabling
  - DMA disabling on RX error
  - MSB first on communication line
  - Time out enabling (and if activated, timeout value)
  - Block length
  - Auto-retry counter

The `HAL_SMARTCARD_Init()` API follows the USART synchronous configuration procedures (details for the procedures are available in reference manual).

This section contains the following APIs:

- [`HAL\_SMARTCARD\_Init\(\)`](#)
- [`HAL\_SMARTCARD\_DelInit\(\)`](#)
- [`HAL\_SMARTCARD\_MspInit\(\)`](#)
- [`HAL\_SMARTCARD\_MspDelInit\(\)`](#)

## 45.2.3 IO operation functions

This subsection provides a set of functions allowing to manage the SMARTCARD data transfers.

Smartcard is a single wire half duplex communication protocol. The Smartcard interface is designed to support asynchronous protocol Smartcards as defined in the ISO 7816-3 standard. The USART should be configured as:

- 8 bits plus parity: where M=1 and PCE=1 in the USART\_CR1 register
  - 1.5 stop bits when transmitting and receiving: where STOP=11 in the USART\_CR2 register.
1. There are two modes of transfer:
    - Blocking mode: The communication is performed in polling mode. The HAL status of all data processing is returned by the same function after finishing transfer.
    - Non-Blocking mode: The communication is performed using Interrupts or DMA, the relevant API's return the HAL status. The end of the data processing will be indicated through the dedicated SMARTCARD IRQ when using Interrupt mode or the DMA IRQ when using DMA mode.
    - The HAL\_SMARTCARD\_TxCpltCallback(), HAL\_SMARTCARD\_RxCpltCallback() user callbacks will be executed respectively at the end of the Transmit or Receive process. The HAL\_SMARTCARD\_ErrorCallback() user callback will be executed when a communication error is detected.
  2. Blocking mode APIs are :
    - HAL\_SMARTCARD\_Transmit()
    - HAL\_SMARTCARD\_Receive()
  3. Non Blocking mode APIs with Interrupt are :
    - HAL\_SMARTCARD\_Transmit\_IT()
    - HAL\_SMARTCARD\_Receive\_IT()
    - HAL\_SMARTCARD\_IRQHandler()
  4. Non Blocking mode functions with DMA are :
    - HAL\_SMARTCARD\_Transmit\_DMA()
    - HAL\_SMARTCARD\_Receive\_DMA()
  5. A set of Transfer Complete Callbacks are provided in non Blocking mode:
    - HAL\_SMARTCARD\_TxCpltCallback()
    - HAL\_SMARTCARD\_RxCpltCallback()
    - HAL\_SMARTCARD\_ErrorCallback()
  6. Non-Blocking mode transfers could be aborted using Abort API's :
    - HAL\_SMARTCARD\_Abort()
    - HAL\_SMARTCARD\_AbortTransmit()
    - HAL\_SMARTCARD\_AbortReceive()
    - HAL\_SMARTCARD\_Abort\_IT()
    - HAL\_SMARTCARD\_AbortTransmit\_IT()
    - HAL\_SMARTCARD\_AbortReceive\_IT()
  7. For Abort services based on interrupts (HAL\_SMARTCARD\_Abortxxx\_IT), a set of Abort Complete Callbacks are provided:
    - HAL\_SMARTCARD\_AbortCpltCallback()
    - HAL\_SMARTCARD\_AbortTransmitCpltCallback()
    - HAL\_SMARTCARD\_AbortReceiveCpltCallback()
  8. In Non-Blocking mode transfers, possible errors are split into 2 categories. Errors are handled as follows :
    - Error is considered as Recoverable and non blocking : Transfer could go till end, but error severity is to be evaluated by user : this concerns Frame Error, Parity Error or Noise Error in Interrupt mode reception . Received character is then retrieved and stored in Rx buffer, Error code is set to allow user to identify error type, and HAL\_SMARTCARD\_ErrorCallback() user callback is executed.  
Transfer is kept ongoing on SMARTCARD side. If user wants to abort it, Abort services should be called by user.
    - Error is considered as Blocking : Transfer could not be completed properly and is aborted. This concerns Frame Error in Interrupt mode transmission, Overrun Error in Interrupt mode reception and all errors in DMA mode. Error code is set to allow

user to identify error type, and HAL\_SMARTCARD\_ErrorCallback() user callback is executed.

This section contains the following APIs:

- [`HAL\_SMARTCARD\_Transmit\(\)`](#)
- [`HAL\_SMARTCARD\_Receive\(\)`](#)
- [`HAL\_SMARTCARD\_Transmit\_IT\(\)`](#)
- [`HAL\_SMARTCARD\_Receive\_IT\(\)`](#)
- [`HAL\_SMARTCARD\_Transmit\_DMA\(\)`](#)
- [`HAL\_SMARTCARD\_Receive\_DMA\(\)`](#)
- [`HAL\_SMARTCARD\_Abort\(\)`](#)
- [`HAL\_SMARTCARD\_AbortTransmit\(\)`](#)
- [`HAL\_SMARTCARD\_AbortReceive\(\)`](#)
- [`HAL\_SMARTCARD\_Abort\_IT\(\)`](#)
- [`HAL\_SMARTCARD\_AbortTransmit\_IT\(\)`](#)
- [`HAL\_SMARTCARD\_AbortReceive\_IT\(\)`](#)
- [`HAL\_SMARTCARD\_IRQHandler\(\)`](#)
- [`HAL\_SMARTCARD\_TxCpltCallback\(\)`](#)
- [`HAL\_SMARTCARD\_RxCpltCallback\(\)`](#)
- [`HAL\_SMARTCARD\_ErrorCallback\(\)`](#)
- [`HAL\_SMARTCARD\_AbortCpltCallback\(\)`](#)
- [`HAL\_SMARTCARD\_AbortTransmitCpltCallback\(\)`](#)
- [`HAL\_SMARTCARD\_AbortReceiveCpltCallback\(\)`](#)

#### 45.2.4 Peripheral State and Errors functions

This subsection provides a set of functions allowing to return the State of SmartCard handle and also return Peripheral Errors occurred during communication process

- `HAL_SMARTCARD_GetState()` API can be helpful to check in run-time the state of the SMARTCARD peripheral.
- `HAL_SMARTCARD_GetError()` checks in run-time errors that could occur during communication.

This section contains the following APIs:

- [`HAL\_SMARTCARD\_GetState\(\)`](#)
- [`HAL\_SMARTCARD\_GetError\(\)`](#)

#### 45.2.5 Detailed description of functions

##### `HAL_SMARTCARD_Init`

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_SMARTCARD_Init(SMARTCARD_HandleTypeDef * hsmartcard)</code>                                                                                                               |
| Function description | Initialize the SMARTCARD mode according to the specified parameters in the SMARTCARD_HandleTypeDef and initialize the associated handle.                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                |

**HAL\_SMARTCARD\_DelInit**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_DelInit<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                             |
| Function description | DeInitialize the SMARTCARD peripheral.                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                |

**HAL\_SMARTCARD\_MspInit**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMARTCARD_MspInit<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                          |
| Function description | Initialize the SMARTCARD MSP.                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |

**HAL\_SMARTCARD\_MspDelInit**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMARTCARD_MspDelInit<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                       |
| Function description | DeInitialize the SMARTCARD MSP.                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |

**HAL\_SMARTCARD\_Transmit**

|                      |                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_Transmit<br/>(SMARTCARD_HandleTypeDef * hsmartcard, uint8_t * pData,<br/>uint16_t Size, uint32_t Timeout)</b>                                                                                                                                                                                                       |
| Function description | Send an amount of data in blocking mode.                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be sent.</li> <li>• <b>Timeout:</b> Timeout duration.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                 |

**HAL\_SMARTCARD\_Receive**

|                      |                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_Receive</b><br><b>(SMARTCARD_HandleTypeDef * hsmartcard, uint8_t * pData,</b><br><b>uint16_t Size, uint32_t Timeout)</b>                                                                                                                                                                                                |
| Function description | Receive an amount of data in blocking mode.                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be received.</li> <li>• <b>Timeout:</b> Timeout duration.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                     |

**HAL\_SMARTCARD\_Transmit\_IT**

|                      |                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_Transmit_IT</b><br><b>(SMARTCARD_HandleTypeDef * hsmartcard, uint8_t * pData,</b><br><b>uint16_t Size)</b>                                                                                                                                                             |
| Function description | Send an amount of data in interrupt mode.                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be sent.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                    |

**HAL\_SMARTCARD\_Receive\_IT**

|                      |                                                                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_Receive_IT</b><br><b>(SMARTCARD_HandleTypeDef * hsmartcard, uint8_t * pData,</b><br><b>uint16_t Size)</b>                                                                                                                                                                  |
| Function description | Receive an amount of data in interrupt mode.                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be received.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                        |

**HAL\_SMARTCARD\_Transmit\_DMA**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_Transmit_DMA</b><br><b>(SMARTCARD_HandleTypeDef * hsmartcard, uint8_t * pData,</b><br><b>uint16_t Size)</b>                                                        |
| Function description | Send an amount of data in DMA mode.                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |

- **pData:** pointer to data buffer.
  - **Size:** amount of data to be sent.
- Return values
- **HAL:** status

### **HAL\_SMARTCARD\_Receive\_DMA**

|                      |                                                                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_Receive_DMA<br/>(SMARTCARD_HandleTypeDef * hsmartcard, uint8_t * pData,<br/>uint16_t Size)</b>                                                                                                                                                                             |
| Function description | Receive an amount of data in DMA mode.                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be received.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• The SMARTCARD-associated USART parity is enabled (PCE = 1), the received data contain the parity bit (MSB position).</li> </ul>                                                                                                                                      |

### **HAL\_SMARTCARD\_Abort**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_Abort<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Abort ongoing transfers (blocking mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul>                                                                                                                                                                                                                                                                                                                                |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable SMARTCARD Interrupts (Tx and Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort (in case of transfer in DMA mode)Set handle State to READY</li> <li>• This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.</li> </ul> |

### **HAL\_SMARTCARD\_AbortTransmit**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_AbortTransmit<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                       |
| Function description | Abort ongoing Transmit transfer (blocking mode).                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                  |

- 
- |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Tx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable SMARTCARD Interrupts (Tx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort (in case of transfer in DMA mode)Set handle State to READY</li> <li>• This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.</li> </ul> |
|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### **HAL\_SMARTCARD\_AbortReceive**

- |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_AbortReceive (SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Abort ongoing Receive transfer (blocking mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul>                                                                                                                                                                                                                                                                                                                            |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Rx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable SMARTCARD Interrupts (Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort (in case of transfer in DMA mode)Set handle State to READY</li> <li>• This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.</li> </ul> |

### **HAL\_SMARTCARD\_Abort\_IT**

- |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_Abort_IT (SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description | Abort ongoing transfers (Interrupt mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable SMARTCARD Interrupts (Tx and Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort_IT (in case of transfer in DMA mode)Set handle State to READYAt abort completion, call user abort complete callback</li> <li>• This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).</li> </ul> |

**HAL\_SMARTCARD\_AbortTransmit\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_AbortTransmit_IT<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description | Abort ongoing Transmit transfer (Interrupt mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Tx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable SMARTCARD Interrupts (Tx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort_IT (in case of transfer in DMA mode)Set handle State to READYAt abort completion, call user abort complete callback</li> <li>• This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).</li> </ul> |

**HAL\_SMARTCARD\_AbortReceive\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMARTCARD_AbortReceive_IT<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Abort ongoing Receive transfer (Interrupt mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Rx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable SMARTCARD Interrupts (Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort_IT (in case of transfer in DMA mode)Set handle State to READYAt abort completion, call user abort complete callback</li> <li>• This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).</li> </ul> |

**HAL\_SMARTCARD\_IRQHandler**

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMARTCARD_IRQHandler<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b> |
| Function description | Handle SMARTCARD interrupt requests.                                            |

---

|               |                                                                                                                                                                                                       |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |

### **HAL\_SMARTCARD\_TxCpltCallback**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMARTCARD_TxCpltCallback(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                        |
| Function description | Tx Transfer completed callback.                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |

### **HAL\_SMARTCARD\_RxCpltCallback**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMARTCARD_RxCpltCallback(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                        |
| Function description | Rx Transfer completed callback.                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |

### **HAL\_SMARTCARD\_ErrorCallback**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMARTCARD_ErrorCallback(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                         |
| Function description | SMARTCARD error callback.                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |

### **HAL\_SMARTCARD\_AbortCpltCallback**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMARTCARD_AbortCpltCallback(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                     |
| Function description | SMARTCARD Abort Complete callback.                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |

**HAL\_SMARTCARD\_AbortTransmitCpltCallback**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMARTCARD_AbortTransmitCpltCallback<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                        |
| Function description | SMARTCARD Abort Complete callback.                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |

**HAL\_SMARTCARD\_AbortReceiveCpltCallback**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMARTCARD_AbortReceiveCpltCallback<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                         |
| Function description | SMARTCARD Abort Receive Complete callback.                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |

**HAL\_SMARTCARD\_GetState**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_SMARTCARD_StateTypeDef<br/>HAL_SMARTCARD_GetState (SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                   |
| Function description | Return the SMARTCARD handle state.                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>SMARTCARD:</b> handle state</li> </ul>                                                                                                                    |

**HAL\_SMARTCARD\_GetError**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_SMARTCARD_GetError<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                                                     |
| Function description | Return the SMARTCARD handle error code.                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>SMARTCARD:</b> handle Error Code</li> </ul>                                                                                                               |

## 45.3 SMARTCARD Firmware driver defines

### 45.3.1 SMARTCARD

#### ***SMARTCARD advanced feature initialization type***

|                                             |                                          |
|---------------------------------------------|------------------------------------------|
| SMARTCARD_ADVFEATURE_NO_INIT                | No advanced feature initialization       |
| SMARTCARD_ADVFEATURE_TXINVERT_INIT          | TX pin active level inversion            |
| SMARTCARD_ADVFEATURE_RXINVERT_INIT          | RX pin active level inversion            |
| SMARTCARD_ADVFEATURE_DATAINVERT_INIT        | Binary data inversion                    |
| SMARTCARD_ADVFEATURE_SWAP_INIT              | TX/RX pins swap                          |
| SMARTCARD_ADVFEATURE_RXOVERRUNDISABLE_INIT  | RX overrun disable                       |
| SMARTCARD_ADVFEATURE_DMADISABLEONERROR_INIT | DMA disable on Reception Error           |
| SMARTCARD_ADVFEATURE_MSBFIRST_INIT          | Most significant bit sent/received first |

#### ***SMARTCARD Clock Phase***

|                       |                                                  |
|-----------------------|--------------------------------------------------|
| SMARTCARD_PHASE_1EDGE | SMARTCARD frame phase on first clock transition  |
| SMARTCARD_PHASE_2EDGE | SMARTCARD frame phase on second clock transition |

#### ***SMARTCARD Clock Polarity***

|                         |                               |
|-------------------------|-------------------------------|
| SMARTCARD_POLARITY_LOW  | SMARTCARD frame low polarity  |
| SMARTCARD_POLARITY_HIGH | SMARTCARD frame high polarity |

#### ***SMARTCARD auto retry counter LSB position in CR3 register***

|                               |                                                           |
|-------------------------------|-----------------------------------------------------------|
| SMARTCARD_CR3_SCARCNT_LSB_POS | SMARTCARD auto retry counter LSB position in CR3 register |
|-------------------------------|-----------------------------------------------------------|

#### ***SMARTCARD advanced feature Binary Data inversion***

|                                      |                               |
|--------------------------------------|-------------------------------|
| SMARTCARD_ADVFEATURE_DATAINV_DISABLE | Binary data inversion disable |
| SMARTCARD_ADVFEATURE_DATAINV_ENABLE  | Binary data inversion enable  |

#### ***SMARTCARD advanced feature DMA Disable on Rx Error***

|                                           |                                |
|-------------------------------------------|--------------------------------|
| SMARTCARD_ADVFEATURE_DMA_ENABLEONRXERROR  | DMA enable on Reception Error  |
| SMARTCARD_ADVFEATURE_DMA_DISABLEONRXERROR | DMA disable on Reception Error |

#### ***SMARTCARD Error***

|                          |              |
|--------------------------|--------------|
| HAL_SMARTCARD_ERROR_NONE | No error     |
| HAL_SMARTCARD_ERROR_PE   | Parity error |
| HAL_SMARTCARD_ERROR_NE   | Noise error  |
| HAL_SMARTCARD_ERROR_FE   | frame error  |

---

|                                      |                        |
|--------------------------------------|------------------------|
| <code>HAL_SMARTCARD_ERROR_ORE</code> | Overrun error          |
| <code>HAL_SMARTCARD_ERROR_DMA</code> | DMA transfer error     |
| <code>HAL_SMARTCARD_ERROR_RTO</code> | Receiver TimeOut error |

***SMARTCARD Exported Macros***

`__HAL_SMARTCARD_RESET_HANDLE_STATE`

**Description:**

- Reset SMARTCARD handle states.

**Parameters:**

- `__HANDLE__`: SMARTCARD handle.

**Return value:**

- None

`__HAL_SMARTCARD_FLUSH_DRREGISTER`

**Description:**

- Flush the Smartcard Data registers.

**Parameters:**

- `__HANDLE__`: specifies the SMARTCARD Handle.

**Return value:**

- None

`__HAL_SMARTCARD_CLEAR_FLAG`

**Description:**

- Clear the specified SMARTCARD pending flag.

**Parameters:**

- `__HANDLE__`: specifies the SMARTCARD Handle.
- `__FLAG__`: specifies the flag to check. This parameter can be any combination of the following values:
  - `SMARTCARD_CLEAR_PEF` Parity error clear flag
  - `SMARTCARD_CLEAR_FEF` Framing error clear flag
  - `SMARTCARD_CLEAR_NEF` Noise detected clear flag
  - `SMARTCARD_CLEAR_OREF` OverRun error clear flag
  - `SMARTCARD_CLEAR_IDLEF` Idle line detected clear flag
  - `SMARTCARD_CLEAR_TCF` Transmission complete clear flag
  - `SMARTCARD_CLEAR_RTOF` Receiver timeout clear flag
  - `SMARTCARD_CLEAR_EOBF` End of block clear flag

**Return value:**

- None

`__HAL_SMARTCARD_CLEAR_PE  
FLAG`

**Description:**

- Clear the SMARTCARD PE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the SMARTCARD Handle.

**Return value:**

- None

`__HAL_SMARTCARD_CLEAR_FE  
FLAG`

**Description:**

- Clear the SMARTCARD FE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the SMARTCARD Handle.

**Return value:**

- None

`__HAL_SMARTCARD_CLEAR_NE  
FLAG`

**Description:**

- Clear the SMARTCARD NE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the SMARTCARD Handle.

**Return value:**

- None

`__HAL_SMARTCARD_CLEAR_  
OREFLAG`

**Description:**

- Clear the SMARTCARD ORE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the SMARTCARD Handle.

**Return value:**

- None

`__HAL_SMARTCARD_CLEAR_  
IDLEFLAG`

**Description:**

- Clear the SMARTCARD IDLE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the SMARTCARD Handle.

**Return value:**

- None

[\\_\\_HAL\\_SMARTCARD\\_GET\\_FLAG](#) **Description:**

- Check whether the specified Smartcard flag is set or not.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the SMARTCARD Handle.
- [\\_\\_FLAG\\_\\_](#): specifies the flag to check. This parameter can be one of the following values:
  - [SMARTCARD\\_FLAG\\_RXACK](#) Receive enable acknowledge flag
  - [SMARTCARD\\_FLAG\\_TEACK](#) Transmit enable acknowledge flag
  - [SMARTCARD\\_FLAG\\_BUSY](#) Busy flag
  - [SMARTCARD\\_FLAG\\_EOBF](#) End of block flag
  - [SMARTCARD\\_FLAG\\_RTOF](#) Receiver timeout flag
  - [SMARTCARD\\_FLAG\\_TXE](#) Transmit data register empty flag
  - [SMARTCARD\\_FLAG\\_TC](#) Transmission complete flag
  - [SMARTCARD\\_FLAG\\_RXNE](#) Receive data register not empty flag
  - [SMARTCARD\\_FLAG\\_IDLE](#) Idle line detection flag
  - [SMARTCARD\\_FLAG\\_ORE](#) Overrun error flag
  - [SMARTCARD\\_FLAG\\_NE](#) Noise error flag
  - [SMARTCARD\\_FLAG\\_FE](#) Framing error flag
  - [SMARTCARD\\_FLAG\\_PE](#) Parity error flag

**Return value:**

- The new state of [\\_\\_FLAG\\_\\_](#) (TRUE or FALSE).

[\\_\\_HAL\\_SMARTCARD\\_ENABLE\\_IT](#)**Description:**

- Enable the specified SmartCard interrupt.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the SMARTCARD Handle.
- [\\_\\_INTERRUPT\\_\\_](#): specifies the SMARTCARD interrupt to enable. This parameter can be one of the following values:
  - [SMARTCARD\\_IT\\_EOB](#) End of block interrupt
  - [SMARTCARD\\_IT\\_RTO](#) Receive timeout interrupt
  - [SMARTCARD\\_IT\\_TXE](#) Transmit data register empty interrupt
  - [SMARTCARD\\_IT\\_TC](#) Transmission

- complete interrupt
- SMARTCARD\_IT\_RXNE Receive data register not empty interrupt
- SMARTCARD\_IT\_IDLE Idle line detection interrupt
- SMARTCARD\_IT\_PE Parity error interrupt
- SMARTCARD\_IT\_ERR Error interrupt(frame error, noise error, overrun error)

**Return value:**

- None

`__HAL_SMARTCARD_DISABLE_IT`

**Description:**

- Disable the specified SmartCard interrupt.

**Parameters:**

- `__HANDLE__`: specifies the SMARTCARD Handle.
- `__INTERRUPT__`: specifies the SMARTCARD interrupt to disable. This parameter can be one of the following values:
  - SMARTCARD\_IT\_EOB End of block interrupt
  - SMARTCARD\_IT\_RTO Receive timeout interrupt
  - SMARTCARD\_IT\_TXE Transmit data register empty interrupt
  - SMARTCARD\_IT\_TC Transmission complete interrupt
  - SMARTCARD\_IT\_RXNE Receive data register not empty interrupt
  - SMARTCARD\_IT\_IDLE Idle line detection interrupt
  - SMARTCARD\_IT\_PE Parity error interrupt
  - SMARTCARD\_IT\_ERR Error interrupt(frame error, noise error, overrun error)

**Return value:**

- None

`__HAL_SMARTCARD_GET_IT`

**Description:**

- Check whether the specified SmartCard interrupt has occurred or not.

**Parameters:**

- `__HANDLE__`: specifies the SMARTCARD Handle.
- `__IT__`: specifies the SMARTCARD interrupt to check. This parameter can be one of the following values:
  - SMARTCARD\_IT\_EOB End of block interrupt

- SMARTCARD\_IT\_RTO Receive timeout interrupt
- SMARTCARD\_IT\_TXE Transmit data register empty interrupt
- SMARTCARD\_IT\_TC Transmission complete interrupt
- SMARTCARD\_IT\_RXNE Receive data register not empty interrupt
- SMARTCARD\_IT\_IDLE Idle line detection interrupt
- SMARTCARD\_IT\_ORE Overrun error interrupt
- SMARTCARD\_IT\_NE Noise error interrupt
- SMARTCARD\_IT\_FE Framing error interrupt
- SMARTCARD\_IT\_PE Parity error interrupt

**Return value:**

- The new state of \_\_IT\_\_ (TRUE or FALSE).

`__HAL_SMARTCARD_GET_IT_SOURCE`**Description:**

- Check whether the specified SmartCard interrupt source is enabled or not.

**Parameters:**

- \_\_HANDLE\_\_: specifies the SMARTCARD Handle.
- \_\_IT\_\_: specifies the SMARTCARD interrupt source to check. This parameter can be one of the following values:
  - SMARTCARD\_IT\_EOB End of block interrupt
  - SMARTCARD\_IT\_RTO Receive timeout interrupt
  - SMARTCARD\_IT\_TXE Transmit data register empty interrupt
  - SMARTCARD\_IT\_TC Transmission complete interrupt
  - SMARTCARD\_IT\_RXNE Receive data register not empty interrupt
  - SMARTCARD\_IT\_IDLE Idle line detection interrupt
  - SMARTCARD\_IT\_ERR Framing, overrun or noise error interrupt
  - SMARTCARD\_IT\_PE Parity error interrupt

**Return value:**

- The new state of \_\_IT\_\_ (TRUE or FALSE).

`__HAL_SMARTCARD_CLEAR_IT`**Description:**

- Clear the specified SMARTCARD ISR flag, in setting the proper ICR register flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the SMARTCARD Handle.
- \_\_IT\_CLEAR\_\_: specifies the interrupt clear register flag that needs to be set to clear the corresponding interrupt. This parameter can be one of the following values:
  - SMARTCARD\_CLEAR\_PEF Parity error clear flag
  - SMARTCARD\_CLEAR\_FEF Framing error clear flag
  - SMARTCARD\_CLEAR\_NEF Noise detected clear flag
  - SMARTCARD\_CLEAR\_OREF OverRun error clear flag
  - SMARTCARD\_CLEAR\_IDLEF Idle line detection clear flag
  - SMARTCARD\_CLEAR\_TCF Transmission complete clear flag
  - SMARTCARD\_CLEAR\_RTOF Receiver timeout clear flag
  - SMARTCARD\_CLEAR\_EOBF End of block clear flag

**Return value:**

- None

[\\_\\_HAL\\_SMARTCARD\\_SEND\\_REQ](#)**Description:**

- Set a specific SMARTCARD request flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the SMARTCARD Handle.
- \_\_REQ\_\_: specifies the request flag to set. This parameter can be one of the following values:
  - SMARTCARD\_RXDATA\_FLUSH\_REQ ST Receive data flush Request
  - SMARTCARD\_TXDATA\_FLUSH\_REQ ST Transmit data flush Request

**Return value:**

- None

[\\_\\_HAL\\_SMARTCARD\\_ONE\\_BIT\\_SAMPLE\\_ENABLE](#)**Description:**

- Enable the SMARTCARD one bit sample method.

**Parameters:**

- \_\_HANDLE\_\_: specifies the SMARTCARD Handle.

**Return value:**

- None

|                                                     |                                                                                                                                                                                                                                                                                                                                                            |
|-----------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>__HAL_SMARTCARD_ONE_BIT_SAMPLE_DISABLE</code> | <p><b>Description:</b></p> <ul style="list-style-type: none"> <li>Disable the SMARTCARD one bit sample method.</li> </ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"> <li><code>__HANDLE__</code>: specifies the SMARTCARD Handle.</li> </ul> <p><b>Return value:</b></p> <ul style="list-style-type: none"> <li>None</li> </ul>          |
| <code>__HAL_SMARTCARD_ENABLE</code>                 | <p><b>Description:</b></p> <ul style="list-style-type: none"> <li>Enable the USART associated to the SMARTCARD Handle.</li> </ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"> <li><code>__HANDLE__</code>: specifies the SMARTCARD Handle.</li> </ul> <p><b>Return value:</b></p> <ul style="list-style-type: none"> <li>None</li> </ul>  |
| <code>__HAL_SMARTCARD_DISABLE</code>                | <p><b>Description:</b></p> <ul style="list-style-type: none"> <li>Disable the USART associated to the SMARTCARD Handle.</li> </ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"> <li><code>__HANDLE__</code>: specifies the SMARTCARD Handle.</li> </ul> <p><b>Return value:</b></p> <ul style="list-style-type: none"> <li>None</li> </ul> |

### **SMARTCARD Flags**

|                                   |                                            |
|-----------------------------------|--------------------------------------------|
| <code>SMARTCARD_FLAG_RXACK</code> | SMARTCARD receive enable acknowledge flag  |
| <code>SMARTCARD_FLAG_TEACK</code> | SMARTCARD transmit enable acknowledge flag |
| <code>SMARTCARD_FLAG_BUSY</code>  | SMARTCARD busy flag                        |
| <code>SMARTCARD_FLAG_EOBF</code>  | SMARTCARD end of block flag                |
| <code>SMARTCARD_FLAG_RTOF</code>  | SMARTCARD receiver timeout flag            |
| <code>SMARTCARD_FLAG_TXE</code>   | SMARTCARD transmit data register empty     |
| <code>SMARTCARD_FLAG_TC</code>    | SMARTCARD transmission complete            |
| <code>SMARTCARD_FLAG_RXNE</code>  | SMARTCARD read data register not empty     |
| <code>SMARTCARD_FLAG_IDLE</code>  | SMARTCARD idle line detection              |
| <code>SMARTCARD_FLAG_ORE</code>   | SMARTCARD overrun error                    |
| <code>SMARTCARD_FLAG_NE</code>    | SMARTCARD noise error                      |
| <code>SMARTCARD_FLAG_FE</code>    | SMARTCARD frame error                      |
| <code>SMARTCARD_FLAG_PE</code>    | SMARTCARD parity error                     |

***SMARTCARD guard time value LSB position in GTPR register***

`SMARTCARD_GTPR_GT_LSB_POS` SMARTCARD guard time value LSB position in GTPR register

***SMARTCARD interruptions flags mask***

`SMARTCARD_IT_MASK` SMARTCARD interruptions flags mask

***SMARTCARD Interrupts Definition***

|                                |                                                     |
|--------------------------------|-----------------------------------------------------|
| <code>SMARTCARD_IT_PE</code>   | SMARTCARD parity error interruption                 |
| <code>SMARTCARD_IT_TXE</code>  | SMARTCARD transmit data register empty interruption |
| <code>SMARTCARD_IT_TC</code>   | SMARTCARD transmission complete interruption        |
| <code>SMARTCARD_IT_RXNE</code> | SMARTCARD read data register not empty interruption |
| <code>SMARTCARD_IT_IDLE</code> | SMARTCARD idle line detection interruption          |
| <code>SMARTCARD_IT_ERR</code>  | SMARTCARD error interruption                        |
| <code>SMARTCARD_IT_ORE</code>  | SMARTCARD overrun error interruption                |
| <code>SMARTCARD_IT_NE</code>   | SMARTCARD noise error interruption                  |
| <code>SMARTCARD_IT_FE</code>   | SMARTCARD frame error interruption                  |
| <code>SMARTCARD_IT_EOB</code>  | SMARTCARD end of block interruption                 |
| <code>SMARTCARD_IT_RTO</code>  | SMARTCARD receiver timeout interruption             |

***SMARTCARD Interruption Clear Flags***

|                                    |                                            |
|------------------------------------|--------------------------------------------|
| <code>SMARTCARD_CLEAR_PEF</code>   | SMARTCARD parity error clear flag          |
| <code>SMARTCARD_CLEAR_FEF</code>   | SMARTCARD framing error clear flag         |
| <code>SMARTCARD_CLEAR_NEF</code>   | SMARTCARD noise detected clear flag        |
| <code>SMARTCARD_CLEAR_OREF</code>  | SMARTCARD overrun error clear flag         |
| <code>SMARTCARD_CLEAR_IDLEF</code> | SMARTCARD idle line detected clear flag    |
| <code>SMARTCARD_CLEAR_TCF</code>   | SMARTCARD transmission complete clear flag |
| <code>SMARTCARD_CLEAR_RTOF</code>  | SMARTCARD receiver time out clear flag     |
| <code>SMARTCARD_CLEAR_EOBF</code>  | SMARTCARD end of block clear flag          |

***SMARTCARD Last Bit***

|                                        |                                                                  |
|----------------------------------------|------------------------------------------------------------------|
| <code>SMARTCARD_LASTBIT_DISABLE</code> | SMARTCARD frame last data bit clock pulse not output to SCLK pin |
| <code>SMARTCARD_LASTBIT_ENABLE</code>  | SMARTCARD frame last data bit clock pulse output to SCLK pin     |

***SMARTCARD Transfer Mode***

|                                   |                          |
|-----------------------------------|--------------------------|
| <code>SMARTCARD_MODE_RX</code>    | SMARTCARD RX mode        |
| <code>SMARTCARD_MODE_TX</code>    | SMARTCARD TX mode        |
| <code>SMARTCARD_MODE_TX_RX</code> | SMARTCARD RX and TX mode |

***SMARTCARD advanced feature MSB first***

|                                                    |                                                  |
|----------------------------------------------------|--------------------------------------------------|
| <code>SMARTCARD_ADVFEATURE_MSBFIRST_DISABLE</code> | Most significant bit sent/received first disable |
|----------------------------------------------------|--------------------------------------------------|

SMARTCARD\_ADVFEATURE\_MSBFIRST\_ENABLE Most significant bit sent/received first enable

**SMARTCARD NACK Enable**

SMARTCARD\_NACK\_ENABLE SMARTCARD NACK transmission disabled

SMARTCARD\_NACK\_DISABLE SMARTCARD NACK transmission enabled

**SMARTCARD One Bit Sampling Method**

SMARTCARD\_ONE\_BIT\_SAMPLE\_DISABLE SMARTCARD frame one-bit sample disabled

SMARTCARD\_ONE\_BIT\_SAMPLE\_ENABLE SMARTCARD frame one-bit sample enabled

**SMARTCARD advanced feature Overrun Disable**

SMARTCARD\_ADVFEATURE\_OVERRUN\_ENABLE RX overrun enable

SMARTCARD\_ADVFEATURE\_OVERRUN\_DISABLE RX overrun disable

**SMARTCARD Parity**

SMARTCARD\_PARITY\_EVEN SMARTCARD frame even parity

SMARTCARD\_PARITY\_ODD SMARTCARD frame odd parity

**SMARTCARD Request Parameters**

SMARTCARD\_RXDATA\_FLUSH\_REQUEST Receive data flush request

SMARTCARD\_TXDATA\_FLUSH\_REQUEST Transmit data flush request

**SMARTCARD block length LSB position in RTOR register**

SMARTCARD\_RTOR\_BLEN\_LSB\_POS SMARTCARD block length LSB position in RTOR register

**SMARTCARD advanced feature RX pin active level inversion**

SMARTCARD\_ADVFEATURE\_RXINV\_DISABLE RX pin active level inversion disable

SMARTCARD\_ADVFEATURE\_RXINV\_ENABLE RX pin active level inversion enable

**SMARTCARD advanced feature RX TX pins swap**

SMARTCARD\_ADVFEATURE\_SWAP\_DISABLE TX/RX pins swap disable

SMARTCARD\_ADVFEATURE\_SWAP\_ENABLE TX/RX pins swap enable

**SMARTCARD Number of Stop Bits**

SMARTCARD\_STOPBITS\_0\_5 SMARTCARD frame with 0.5 stop bit

SMARTCARD\_STOPBITS\_1\_5 SMARTCARD frame with 1.5 stop bits

**SMARTCARD Timeout Enable**

SMARTCARD\_TIMEOUT\_DISABLE SMARTCARD receiver timeout disabled

SMARTCARD\_TIMEOUT\_ENABLE SMARTCARD receiver timeout enabled

**SMARTCARD advanced feature TX pin active level inversion**

SMARTCARD\_ADVFEATURE\_TXINV\_DISABLE TX pin active level inversion disable

SMARTCARD\_ADVFEATURE\_TXINV\_ENABLE TX pin active level inversion enable

***SMARTCARD Word Length***

SMARTCARD\_WORDLENGTH\_9B SMARTCARD frame length

## 46 HAL SMARTCARD Extension Driver

### 46.1 SMARTCARDEEx Firmware driver API description

#### 46.1.1 SMARTCARD peripheral extended features

The Extended SMARTCARD HAL driver can be used as follows:

1. After having configured the SMARTCARD basic features with `HAL_SMARTCARD_Init()`, then program SMARTCARD advanced features if required (TX/RX pins swap, TimeOut, auto-retry counter,...) in the `hsmartcard.AdvancedInit` structure.

#### 46.1.2 Peripheral Control functions

This subsection provides a set of functions allowing to initialize the SMARTCARD.

- `HAL_SMARTCARDEEx_BlockLength_Config()` API allows to configure the Block Length on the fly
- `HAL_SMARTCARDEEx_TimeOut_Config()` API allows to configure the receiver timeout value on the fly
- `HAL_SMARTCARDEEx_EnableReceiverTimeOut()` API enables the receiver timeout feature
- `HAL_SMARTCARDEEx_DisableReceiverTimeOut()` API disables the receiver timeout feature

This section contains the following APIs:

- `HAL_SMARTCARDEEx_BlockLength_Config()`
- `HAL_SMARTCARDEEx_TimeOut_Config()`
- `HAL_SMARTCARDEEx_EnableReceiverTimeOut()`
- `HAL_SMARTCARDEEx_DisableReceiverTimeOut()`

#### 46.1.3 Detailed description of functions

##### `HAL_SMARTCARDEEx_BlockLength_Config`

|                      |                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void HAL_SMARTCARDEEx_BlockLength_Config(SMARTCARD_HandleTypeDef * hsmartcard, uint8_t BlockLength)</code>                                                                                                                                                              |
| Function description | Update on the fly the SMARTCARD block length in RTOR register.                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li><li>• <b>BlockLength:</b> SMARTCARD block length (8-bit long at most)</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                                                                                                                                 |

**HAL\_SMARTCARDEEx\_TimeOut\_Config**

|                      |                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMARTCARDEEx_TimeOut_Config<br/>(SMARTCARD_HandleTypeDef * hsmartcard, uint32_t<br/>TimeOutValue)</b>                                                                                                                                                                                                                             |
| Function description | Update on the fly the receiver timeout value in RTOR register.                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> <li>• <b>TimeOutValue:</b> receiver timeout value in number of baud blocks. The timeout value must be less or equal to 0xFFFFFFFF.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                               |

**HAL\_SMARTCARDEEx\_EnableReceiverTimeOut**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef<br/>HAL_SMARTCARDEEx_EnableReceiverTimeOut<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                        |
| Function description | Enable the SMARTCARD receiver timeout feature.                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                |

**HAL\_SMARTCARDEEx\_DisableReceiverTimeOut**

|                      |                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef<br/>HAL_SMARTCARDEEx_DisableReceiverTimeOut<br/>(SMARTCARD_HandleTypeDef * hsmartcard)</b>                                                                                       |
| Function description | Disable the SMARTCARD receiver timeout feature.                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmartcard:</b> Pointer to a SMARTCARD_HandleTypeDef structure that contains the configuration information for the specified SMARTCARD module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                |

## 47 HAL SMBUS Generic Driver

### 47.1 SMBUS Firmware driver registers structures

#### 47.1.1 SMBUS\_InitTypeDef

##### Data Fields

- *uint32\_t Timing*
- *uint32\_t AnalogFilter*
- *uint32\_t OwnAddress1*
- *uint32\_t AddressingMode*
- *uint32\_t DualAddressMode*
- *uint32\_t OwnAddress2*
- *uint32\_t OwnAddress2Masks*
- *uint32\_t GeneralCallMode*
- *uint32\_t NoStretchMode*
- *uint32\_t PacketErrorCheckMode*
- *uint32\_t PeripheralMode*
- *uint32\_t SMBusTimeout*

##### Field Documentation

- ***uint32\_t SMBUS\_InitTypeDef::Timing***  
Specifies the SMBUS\_TIMINGR\_register value. This parameter calculated by referring to SMBUS initialization section in Reference manual
- ***uint32\_t SMBUS\_InitTypeDef::AnalogFilter***  
Specifies if Analog Filter is enable or not. This parameter can be a value of [\*\*SMBUS\\_Analog\\_Filter\*\*](#)
- ***uint32\_t SMBUS\_InitTypeDef::OwnAddress1***  
Specifies the first device own address. This parameter can be a 7-bit or 10-bit address.
- ***uint32\_t SMBUS\_InitTypeDef::AddressingMode***  
Specifies if 7-bit or 10-bit addressing mode for master is selected. This parameter can be a value of [\*\*SMBUS\\_addressing\\_mode\*\*](#)
- ***uint32\_t SMBUS\_InitTypeDef::DualAddressMode***  
Specifies if dual addressing mode is selected. This parameter can be a value of [\*\*SMBUS\\_dual\\_addressing\\_mode\*\*](#)
- ***uint32\_t SMBUS\_InitTypeDef::OwnAddress2***  
Specifies the second device own address if dual addressing mode is selected This parameter can be a 7-bit address.
- ***uint32\_t SMBUS\_InitTypeDef::OwnAddress2Masks***  
Specifies the acknoledge mask address second device own address if dual addressing mode is selected This parameter can be a value of [\*\*SMBUS\\_own\\_address2\\_masks\*\*](#).
- ***uint32\_t SMBUS\_InitTypeDef::GeneralCallMode***  
Specifies if general call mode is selected. This parameter can be a value of [\*\*SMBUS\\_general\\_call\\_addressing\\_mode\*\*](#).
- ***uint32\_t SMBUS\_InitTypeDef::NoStretchMode***  
Specifies if nostretch mode is selected. This parameter can be a value of [\*\*SMBUS\\_nostretch\\_mode\*\*](#)

- ***uint32\_t SMBUS\_InitTypeDef::PacketErrorCheckMode***  
Specifies if Packet Error Check mode is selected. This parameter can be a value of **SMBUS\_packet\_error\_check\_mode**
- ***uint32\_t SMBUS\_InitTypeDef::PeripheralMode***  
Specifies which mode of Peripheral is selected. This parameter can be a value of **SMBUS\_peripheral\_mode**
- ***uint32\_t SMBUS\_InitTypeDef::SMBusTimeout***  
Specifies the content of the 32 Bits SMBUS\_TIMEOUT\_register value. (Enable bits and different timeout values) This parameter calculated by referring to SMBUS initialization section in Reference manual

#### 47.1.2 SMBUS\_HandleTypeDef

##### Data Fields

- ***I2C\_TypeDef \* Instance***
- ***SMBUS\_InitTypeDef Init***
- ***uint8\_t \* pBuffPtr***
- ***uint16\_t XferSize***
- ***\_\_IO uint16\_t XferCount***
- ***\_\_IO uint32\_t XferOptions***
- ***\_\_IO uint32\_t PreviousState***
- ***HAL\_LockTypeDef Lock***
- ***\_\_IO uint32\_t State***
- ***\_\_IO uint32\_t ErrorCode***

##### Field Documentation

- ***I2C\_TypeDef\* SMBUS\_HandleTypeDef::Instance***  
SMBUS registers base address
- ***SMBUS\_InitTypeDef SMBUS\_HandleTypeDef::Init***  
SMBUS communication parameters
- ***uint8\_t\* SMBUS\_HandleTypeDef::pBuffPtr***  
Pointer to SMBUS transfer buffer
- ***uint16\_t SMBUS\_HandleTypeDef::XferSize***  
SMBUS transfer size
- ***\_\_IO uint16\_t SMBUS\_HandleTypeDef::XferCount***  
SMBUS transfer counter
- ***\_\_IO uint32\_t SMBUS\_HandleTypeDef::XferOptions***  
SMBUS transfer options
- ***\_\_IO uint32\_t SMBUS\_HandleTypeDef::PreviousState***  
SMBUS communication Previous state
- ***HAL\_LockTypeDef SMBUS\_HandleTypeDef::Lock***  
SMBUS locking object
- ***\_\_IO uint32\_t SMBUS\_HandleTypeDef::State***  
SMBUS communication state
- ***\_\_IO uint32\_t SMBUS\_HandleTypeDef::ErrorCode***  
SMBUS Error code

## 47.2 SMBUS Firmware driver API description

### 47.2.1 How to use this driver

The SMBUS HAL driver can be used as follows:

1. Declare a SMBUS\_HandleTypeDef handle structure, for example:  
SMBUS\_HandleTypeDef hsmbus;
2. Initialize the SMBUS low level resources by implementing the HAL\_SMBUS\_MspInit() API:
  - a. Enable the SMBUSx interface clock
  - b. SMBUS pins configuration
    - Enable the clock for the SMBUS GPIOs
    - Configure SMBUS pins as alternate function open-drain
  - c. NVIC configuration if you need to use interrupt process
    - Configure the SMBUSx interrupt priority
    - Enable the NVIC SMBUS IRQ Channel
3. Configure the Communication Clock Timing, Bus Timeout, Own Address1, Master Addressing mode, Dual Addressing mode, Own Address2, Own Address2 Mask, General call, Nostretch mode, Peripheral mode and Packet Error Check mode in the hsmbus Init structure.
4. Initialize the SMBUS registers by calling the HAL\_SMBUS\_Init() API:
  - These API's configures also the low level Hardware GPIO, CLOCK, CORTEX...etc) by calling the customized HAL\_SMBUS\_MspInit(&hsmbus) API.
5. To check if target device is ready for communication, use the function HAL\_SMBUS\_IsDeviceReady()
6. For SMBUS IO operations, only one mode of operations is available within this driver

#### Interrupt mode IO operation

- Transmit in master/host SMBUS mode an amount of data in non-blocking mode using HAL\_SMBUS\_Master\_Transmit\_IT()
  - At transmission end of transfer HAL\_SMBUS\_MasterTxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_SMBUS\_MasterTxCpltCallback()
- Receive in master/host SMBUS mode an amount of data in non-blocking mode using HAL\_SMBUS\_Master\_Receive\_IT()
  - At reception end of transfer HAL\_SMBUS\_MasterRxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_SMBUS\_MasterRxCpltCallback()
- Abort a master/host SMBUS process communication with Interrupt using HAL\_SMBUS\_Master\_Abort\_IT()
  - The associated previous transfer callback is called at the end of abort process
  - mean HAL\_SMBUS\_MasterTxCpltCallback() in case of previous state was master transmit
  - mean HAL\_SMBUS\_MasterRxCpltCallback() in case of previous state was master receive
- Enable/disable the Address listen mode in slave/device or host/slave SMBUS mode using HAL\_SMBUS\_EnableListen\_IT() HAL\_SMBUS\_DisableListen\_IT()
  - When address slave/device SMBUS match, HAL\_SMBUS\_AddrCallback() is executed and user can add his own code to check the Address Match Code and the transmission direction request by master/host (Write/Read).

- At Listen mode end HAL\_SMBUS\_ListenCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_SMBUS\_ListenCpltCallback()
- Transmit in slave/device SMBUS mode an amount of data in non-blocking mode using HAL\_SMBUS\_Slave\_Transmit\_IT()
  - At transmission end of transfer HAL\_SMBUS\_SlaveTxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_SMBUS\_SlaveTxCpltCallback()
- Receive in slave/device SMBUS mode an amount of data in non-blocking mode using HAL\_SMBUS\_Slave\_Receive\_IT()
  - At reception end of transfer HAL\_SMBUS\_SlaveRxCpltCallback() is executed and user can add his own code by customization of function pointer HAL\_SMBUS\_SlaveRxCpltCallback()
- Enable/Disable the SMBUS alert mode using HAL\_SMBUS\_EnableAlert\_IT()
  - When SMBUS Alert is generated HAL\_SMBUS\_ErrorCallback() is executed and user can add his own code by customization of function pointer HAL\_SMBUS\_ErrorCallback() to check the Alert Error Code using function HAL\_SMBUS\_GetError()
- Get HAL state machine or error values using HAL\_SMBUS\_GetState() or HAL\_SMBUS\_GetError()
- In case of transfer Error, HAL\_SMBUS\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_SMBUS\_ErrorCallback() to check the Error Code using function HAL\_SMBUS\_GetError()

### SMBUS HAL driver macros list

Below the list of most used macros in SMBUS HAL driver.

- \_\_HAL\_SMBUS\_ENABLE: Enable the SMBUS peripheral
- \_\_HAL\_SMBUS\_DISABLE: Disable the SMBUS peripheral
- \_\_HAL\_SMBUS\_GET\_FLAG: Check whether the specified SMBUS flag is set or not
- \_\_HAL\_SMBUS\_CLEAR\_FLAG: Clear the specified SMBUS pending flag
- \_\_HAL\_SMBUS\_ENABLE\_IT: Enable the specified SMBUS interrupt
- \_\_HAL\_SMBUS\_DISABLE\_IT: Disable the specified SMBUS interrupt



You can refer to the SMBUS HAL driver header file for more useful macros

#### 47.2.2 Initialization and de-initialization functions

This subsection provides a set of functions allowing to initialize and deinitialize the SMBUSx peripheral:

- User must Implement HAL\_SMBUS\_MspInit() function in which he configures all related peripherals resources (CLOCK, GPIO, IT and NVIC ).
- Call the function HAL\_SMBUS\_Init() to configure the selected device with the selected configuration:
  - Clock Timing
  - Bus Timeout
  - Analog Filter mode
  - Own Address 1

- Addressing mode (Master, Slave)
- Dual Addressing mode
- Own Address 2
- Own Address 2 Mask
- General call mode
- Nostretch mode
- Packet Error Check mode
- Peripheral mode
- Call the function HAL\_SMBUS\_DelInit() to restore the default configuration of the selected SMBUSx peripheral.
- Enable/Disable Analog/Digital filters with HAL\_SMBUS\_ConfigAnalogFilter() and HAL\_SMBUS\_ConfigDigitalFilter().

This section contains the following APIs:

- [\*\*\*HAL\\_SMBUS\\_Init\(\)\*\*\*](#)
- [\*\*\*HAL\\_SMBUS\\_DelInit\(\)\*\*\*](#)
- [\*\*\*HAL\\_SMBUS\\_MspInit\(\)\*\*\*](#)
- [\*\*\*HAL\\_SMBUS\\_MspDelInit\(\)\*\*\*](#)
- [\*\*\*HAL\\_SMBUS\\_ConfigAnalogFilter\(\)\*\*\*](#)
- [\*\*\*HAL\\_SMBUS\\_ConfigDigitalFilter\(\)\*\*\*](#)

### 47.2.3 IO operation functions

This subsection provides a set of functions allowing to manage the SMBUS data transfers.

1. Blocking mode function to check if device is ready for usage is :
  - HAL\_SMBUS\_IsDeviceReady()
2. There is only one mode of transfer:
  - Non-Blocking mode : The communication is performed using Interrupts. These functions return the status of the transfer startup. The end of the data processing will be indicated through the dedicated SMBUS IRQ when using Interrupt mode.
3. Non-Blocking mode functions with Interrupt are :
  - HAL\_SMBUS\_Master\_Transmit\_IT()
  - HAL\_SMBUS\_Master\_Receive\_IT()
  - HAL\_SMBUS\_Slave\_Transmit\_IT()
  - HAL\_SMBUS\_Slave\_Receive\_IT()
  - HAL\_SMBUS\_EnableListen\_IT() or alias HAL\_SMBUS\_EnableListen\_IT()
  - HAL\_SMBUS\_DisableListen\_IT()
  - HAL\_SMBUS\_EnableAlert\_IT()
  - HAL\_SMBUS\_DisableAlert\_IT()
4. A set of Transfer Complete Callbacks are provided in non-Blocking mode:
  - HAL\_SMBUS\_MasterTxCpltCallback()
  - HAL\_SMBUS\_MasterRxCpltCallback()
  - HAL\_SMBUS\_SlaveTxCpltCallback()
  - HAL\_SMBUS\_SlaveRxCpltCallback()
  - HAL\_SMBUS\_AddrCallback()
  - HAL\_SMBUS\_ListenCpltCallback()
  - HAL\_SMBUS\_ErrorCallback()

This section contains the following APIs:

- [\*\*\*HAL\\_SMBUS\\_Master\\_Transmit\\_IT\(\)\*\*\*](#)
- [\*\*\*HAL\\_SMBUS\\_Master\\_Receive\\_IT\(\)\*\*\*](#)
- [\*\*\*HAL\\_SMBUS\\_Master\\_Abort\\_IT\(\)\*\*\*](#)
- [\*\*\*HAL\\_SMBUS\\_Slave\\_Transmit\\_IT\(\)\*\*\*](#)

- [\*HAL\\_SMBUS\\_Slave\\_Receive\\_IT\(\)\*](#)
- [\*HAL\\_SMBUS\\_EnableListen\\_IT\(\)\*](#)
- [\*HAL\\_SMBUS\\_DisableListen\\_IT\(\)\*](#)
- [\*HAL\\_SMBUS\\_EnableAlert\\_IT\(\)\*](#)
- [\*HAL\\_SMBUS\\_DisableAlert\\_IT\(\)\*](#)
- [\*HAL\\_SMBUS\\_IsDeviceReady\(\)\*](#)

#### 47.2.4 Peripheral State and Errors functions

This subsection permits to get in run-time the status of the peripheral and the data flow.

This section contains the following APIs:

- [\*HAL\\_SMBUS\\_GetState\(\)\*](#)
- [\*HAL\\_SMBUS\\_GetError\(\)\*](#)

#### 47.2.5 Detailed description of functions

##### **HAL\_SMBUS\_Init**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_Init<br/>(SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                         |
| Function description | Initialize the SMBUS according to the specified parameters in the SMBUS_InitTypeDef and initialize the associated handle.                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                             |

##### **HAL\_SMBUS\_DelInit**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_DelInit<br/>(SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                      |
| Function description | DeInitialize the SMBUS peripheral.                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                             |

##### **HAL\_SMBUS\_MspInit**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_MspInit (SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                                       |
| Function description | Initialize the SMBUS MSP.                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                    |

**HAL\_SMBUS\_MspDeInit**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_MspDeInit (SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                                     |
| Function description | DeInitialize the SMBUS MSP.                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                    |

**HAL\_SMBUS\_ConfigAnalogFilter**

|                      |                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_ConfigAnalogFilter (SMBUS_HandleTypeDef * hsmbus, uint32_t AnalogFilter)</b>                                                                                                                                                                                                                                                                                |
| Function description | Configure Analog noise filter.                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> <li>• <b>AnalogFilter:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– SMBUS_ANALOGFILTER_ENABLE</li> <li>– SMBUS_ANALOGFILTER_DISABLE</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                     |

**HAL\_SMBUS\_ConfigDigitalFilter**

|                      |                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_ConfigDigitalFilter (SMBUS_HandleTypeDef * hsmbus, uint32_t DigitalFilter)</b>                                                                                                                                                                                    |
| Function description | Configure Digital noise filter.                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> <li>• <b>DigitalFilter:</b> Coefficient of digital noise filter between Min_Data=0x00 and Max_Data=0x0F.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                           |

**HAL\_SMBUS\_IsDeviceReady**

|                      |                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_IsDeviceReady (SMBUS_HandleTypeDef * hsmbus, uint16_t DevAddress, uint32_t Trials, uint32_t Timeout)</b>                                                                                                                                                                                                                                      |
| Function description | Check if target device is ready for communication.                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>Trials:</b> Number of trials</li> </ul> |

- **Timeout:** Timeout duration
- Return values
- **HAL:** status

### **HAL\_SMBUS\_Master\_Transmit\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_Master_Transmit_IT<br/>(SMBUS_HandleTypeDef * hsmbus, uint16_t DevAddress,<br/>uint8_t * pData, uint16_t Size, uint32_t XferOptions)</b>                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Transmit in master/host SMBUS mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>XferOptions:</b> Options of Transfer, value of SMBUS XferOptions definition</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

### **HAL\_SMBUS\_Master\_Receive\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_Master_Receive_IT<br/>(SMBUS_HandleTypeDef * hsmbus, uint16_t DevAddress,<br/>uint8_t * pData, uint16_t Size, uint32_t XferOptions)</b>                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Receive in master/host SMBUS mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>XferOptions:</b> Options of Transfer, value of SMBUS XferOptions definition</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

**HAL\_SMBUS\_Master\_Abort\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_Master_Abort_IT<br/>(SMBUS_HandleTypeDef * hsmbus, uint16_t DevAddress)</b>                                                                                                                                                                                                                        |
| Function description | Abort a master/host SMBUS process communication with Interrupt.                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> <li>• <b>DevAddress:</b> Target device address: The device 7 bits address value in datasheet must be shift at right before call interface</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                            |
| Notes                | <ul style="list-style-type: none"> <li>• This abort can be called only if state is ready</li> </ul>                                                                                                                                                                                                                               |

**HAL\_SMBUS\_Slave\_Transmit\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_Slave_Transmit_IT<br/>(SMBUS_HandleTypeDef * hsmbus, uint8_t * pData, uint16_t Size, uint32_t XferOptions)</b>                                                                                                                                                                                                                                  |
| Function description | Transmit in slave/device SMBUS mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>XferOptions:</b> Options of Transfer, value of SMBUS XferOptions definition</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                         |

**HAL\_SMBUS\_Slave\_Receive\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_Slave_Receive_IT<br/>(SMBUS_HandleTypeDef * hsmbus, uint8_t * pData, uint16_t Size, uint32_t XferOptions)</b>                                                                                                                                                                                                                                   |
| Function description | Receive in slave/device SMBUS mode an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> <li>• <b>pData:</b> Pointer to data buffer</li> <li>• <b>Size:</b> Amount of data to be sent</li> <li>• <b>XferOptions:</b> Options of Transfer, value of SMBUS XferOptions definition</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                         |

**HAL\_SMBUS\_EnableAlert\_IT**

|                      |                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_EnableAlert_IT<br/>(SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                           |
| Function description | Enable the SMBUS alert mode with Interrupt.                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUSx peripheral.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                         |

**HAL\_SMBUS\_DisableAlert\_IT**

|                      |                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_DisableAlert_IT<br/>(SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                          |
| Function description | Disable the SMBUS alert mode with Interrupt.                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUSx peripheral.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                         |

**HAL\_SMBUS\_EnableListen\_IT**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_EnableListen_IT<br/>(SMBUS_HandleTypeDef * hsmbus)</b>                                                                                              |
| Function description | Enable the Address listen mode with Interrupt.                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                             |

**HAL\_SMBUS\_DisableListen\_IT**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SMBUS_DisableListen_IT<br/>(SMBUS_HandleTypeDef * hsmbus)</b>                                                                                             |
| Function description | Disable the Address listen mode with Interrupt.                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                             |

**HAL\_SMBUS\_EV\_IRQHandler**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_EV_IRQHandler (SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                             |
| Function description | Handle SMBUS event interrupt request.                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"><li><b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                                                                    |

**HAL\_SMBUS\_ER\_IRQHandler**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_ER_IRQHandler (SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                             |
| Function description | Handle SMBUS error interrupt request.                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"><li><b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                                                                    |

**HAL\_SMBUS\_MasterTxCpltCallback**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_MasterTxCpltCallback (SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                      |
| Function description | Master Tx Transfer completed callback.                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"><li><b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                                                                    |

**HAL\_SMBUS\_MasterRxCpltCallback**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_MasterRxCpltCallback (SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                      |
| Function description | Master Rx Transfer completed callback.                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"><li><b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                                                                    |

**HAL\_SMBUS\_SlaveTxCpltCallback**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_SlaveTxCpltCallback<br/>(SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                       |
| Function description | Slave Tx Transfer completed callback.                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                    |

**HAL\_SMBUS\_SlaveRxCpltCallback**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_SlaveRxCpltCallback<br/>(SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                       |
| Function description | Slave Rx Transfer completed callback.                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                    |

**HAL\_SMBUS\_AddrCallback**

|                      |                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_AddrCallback (SMBUS_HandleTypeDef * hsmbus, uint8_t TransferDirection, uint16_t AddrMatchCode)</b>                                                                                                                                                                                                       |
| Function description | Slave Address Match callback.                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> <li>• <b>TransferDirection:</b> Master request Transfer Direction (Write/Read)</li> <li>• <b>AddrMatchCode:</b> Address Match Code</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                            |

**HAL\_SMBUS\_ListenCpltCallback**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_ListenCpltCallback<br/>(SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                        |
| Function description | Listen Complete callback.                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                    |

**HAL\_SMBUS\_ErrorCallback**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SMBUS_ErrorCallback (SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                                 |
| Function description | SMBUS error callback.                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                    |

**HAL\_SMBUS\_GetState**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_SMBUS_GetState (SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                                  |
| Function description | Return the SMBUS handle state.                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                              |

**HAL\_SMBUS\_GetError**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_SMBUS_GetError (SMBUS_HandleTypeDef * hsmbus)</b>                                                                                                                  |
| Function description | Return the SMBUS error code.                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsmbus:</b> Pointer to a SMBUS_HandleTypeDef structure that contains the configuration information for the specified SMBUS.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>SMBUS:</b> Error Code</li> </ul>                                                                                                       |

## 47.3 SMBUS Firmware driver defines

### 47.3.1 SMBUS

***SMBUS addressing mode***

SMBUS\_ADDRESSINGMODE\_7BIT

SMBUS\_ADDRESSINGMODE\_10BIT

***SMBUS Analog Filter***

SMBUS\_ANALOGFILTER\_ENABLE

SMBUS\_ANALOGFILTER\_DISABLE

***SMBUS dual addressing mode***

SMBUS\_DUALADDRESS\_DISABLE

SMBUS\_DUALADDRESS\_ENABLE

**SMBUS Error Code definition**

|                            |                   |
|----------------------------|-------------------|
| HAL_SMBUS_ERROR_NONE       | No error          |
| HAL_SMBUS_ERROR_BERR       | BERR error        |
| HAL_SMBUS_ERROR_ARLO       | ARLO error        |
| HAL_SMBUS_ERROR_ACKF       | ACKF error        |
| HAL_SMBUS_ERROR_OVR        | OVR error         |
| HAL_SMBUS_ERROR_HALTIMEOUT | Timeout error     |
| HAL_SMBUS_ERROR_BUSTIMEOUT | Bus Timeout error |
| HAL_SMBUS_ERROR_ALERT      | Alert error       |
| HAL_SMBUS_ERROR_PECERR     | PEC error         |

**SMBUS Exported Macros**

`_HAL_SMBUS_RESET_HANDLE_STATE` **Description:**

- Reset SMBUS handle state.

**Parameters:**

- `_HANDLE_`: specifies the SMBUS Handle.

**Return value:**

- None

`_HAL_SMBUS_ENABLE_IT` **Description:**

- Enable the specified SMBUS interrupts.

**Parameters:**

- `_HANDLE_`: specifies the SMBUS Handle.
- `_INTERRUPT_`: specifies the interrupt source to enable. This parameter can be one of the following values:
  - `SMBUS_IT_ERRI` Errors interrupt enable
  - `SMBUS_IT_TCI` Transfer complete interrupt enable
  - `SMBUS_IT_STOPI` STOP detection interrupt enable
  - `SMBUS_IT_NACKI` NACK received interrupt enable
  - `SMBUS_IT_ADDRI` Address match interrupt enable
  - `SMBUS_IT_RXI` RX interrupt enable
  - `SMBUS_IT_TXI` TX interrupt enable

**Return value:**

- None

[\\_\\_HAL\\_SMBUS\\_DISABLE\\_IT](#)**Description:**

- Disable the specified SMBUS interrupts.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the SMBUS Handle.
- [\\_\\_INTERRUPT\\_\\_](#): specifies the interrupt source to disable. This parameter can be one of the following values:
  - [SMBUS\\_IT\\_ERRI](#) Errors interrupt enable
  - [SMBUS\\_IT\\_TCI](#) Transfer complete interrupt enable
  - [SMBUS\\_IT\\_STOPI](#) STOP detection interrupt enable
  - [SMBUS\\_IT\\_NACKI](#) NACK received interrupt enable
  - [SMBUS\\_IT\\_ADDRI](#) Address match interrupt enable
  - [SMBUS\\_IT\\_RXI](#) RX interrupt enable
  - [SMBUS\\_IT\\_TXI](#) TX interrupt enable

**Return value:**

- None

[\\_\\_HAL\\_SMBUS\\_GET\\_IT\\_SOURCE](#)**Description:**

- Check whether the specified SMBUS interrupt source is enabled or not.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the SMBUS Handle.
- [\\_\\_INTERRUPT\\_\\_](#): specifies the SMBUS interrupt source to check. This parameter can be one of the following values:
  - [SMBUS\\_IT\\_ERRI](#) Errors interrupt enable
  - [SMBUS\\_IT\\_TCI](#) Transfer complete interrupt enable
  - [SMBUS\\_IT\\_STOPI](#) STOP detection interrupt enable
  - [SMBUS\\_IT\\_NACKI](#) NACK received interrupt enable
  - [SMBUS\\_IT\\_ADDRI](#) Address match interrupt enable
  - [SMBUS\\_IT\\_RXI](#) RX interrupt enable
  - [SMBUS\\_IT\\_TXI](#) TX interrupt

enable

**Return value:**

- The: new state of \_\_IT\_\_ (TRUE or FALSE).

SMBUS\_FLAG\_MASK

**Description:**

- Check whether the specified SMBUS flag is set or not.

**Parameters:**

- \_\_HANDLE\_\_: specifies the SMBUS Handle.
- \_\_FLAG\_\_: specifies the flag to check. This parameter can be one of the following values:
  - SMBUS\_FLAG\_TXE Transmit data register empty
  - SMBUS\_FLAG\_TXIS Transmit interrupt status
  - SMBUS\_FLAG\_RXNE Receive data register not empty
  - SMBUS\_FLAG\_ADDR Address matched (slave mode)
  - SMBUS\_FLAG\_AF NACK received flag
  - SMBUS\_FLAG\_STOPF STOP detection flag
  - SMBUS\_FLAG\_TC Transfer complete (master mode)
  - SMBUS\_FLAG\_TCR Transfer complete reload
  - SMBUS\_FLAG\_BERR Bus error
  - SMBUS\_FLAG\_ARLO Arbitration lost
  - SMBUS\_FLAG\_OVR Overrun/Underrun
  - SMBUS\_FLAG\_PECERR PEC error in reception
  - SMBUS\_FLAG\_TIMEOUT Timeout or Tlow detection flag
  - SMBUS\_FLAG\_ALERT SMBus alert
  - SMBUS\_FLAG\_BUSY Bus busy
  - SMBUS\_FLAG\_DIR Transfer direction (slave mode)

**Return value:**

- The: new state of \_\_FLAG\_\_ (TRUE or FALSE).

`__HAL_SMBUS_GET_FLAG`  
`__HAL_SMBUS_CLEAR_FLAG`

**Description:**

- Clear the SMBUS pending flags which are cleared by writing 1 in a specific bit.

**Parameters:**

- `__HANDLE__`: specifies the SMBUS Handle.
- `__FLAG__`: specifies the flag to clear. This parameter can be any combination of the following values:
  - `SMBUS_FLAG_ADDR` Address matched (slave mode)
  - `SMBUS_FLAG_AF_NACK` received flag
  - `SMBUS_FLAG_STOPF` STOP detection flag
  - `SMBUS_FLAG_BERR` Bus error
  - `SMBUS_FLAG_ARLO` Arbitration lost
  - `SMBUS_FLAG_OVR` Overrun/Underrun
  - `SMBUS_FLAG_PECERR` PEC error in reception
  - `SMBUS_FLAG_TIMEOUT` Timeout or Tlow detection flag
  - `SMBUS_FLAG_ALERT` SMBus alert

**Return value:**

- None

`__HAL_SMBUS_ENABLE`

**Description:**

- Enable the specified SMBUS peripheral.

**Parameters:**

- `__HANDLE__`: specifies the SMBUS Handle.

**Return value:**

- None

`__HAL_SMBUS_DISABLE`

**Description:**

- Disable the specified SMBUS peripheral.

**Parameters:**

- `__HANDLE__`: specifies the SMBUS Handle.

**Return value:**

- None

`__HAL_SMBUS_GENERATE_NACK`

**Description:**

- Generate a Non-Acknowledge SMBUS peripheral in Slave mode.

**Parameters:**

- `__HANDLE__`: specifies the SMBUS Handle.

**Return value:**

- None

***SMBUS Flag definition***

`SMBUS_FLAG_TXE`

`SMBUS_FLAG_TXIS`

`SMBUS_FLAG_RXNE`

`SMBUS_FLAG_ADDR`

`SMBUS_FLAG_AF`

`SMBUS_FLAG_STOPF`

`SMBUS_FLAG_TC`

`SMBUS_FLAG_TCR`

`SMBUS_FLAG_BERR`

`SMBUS_FLAG_ARLO`

`SMBUS_FLAG_OVR`

`SMBUS_FLAG_PECERR`

`SMBUS_FLAG_TIMEOUT`

`SMBUS_FLAG_ALERT`

`SMBUS_FLAG_BUSY`

`SMBUS_FLAG_DIR`

***SMBUS general call addressing mode***

`SMBUS_GENERALCALL_DISABLE`

`SMBUS_GENERALCALL_ENABLE`

***SMBUS Interrupt configuration definition***

`SMBUS_IT_ERRI`

`SMBUS_IT_TCI`

`SMBUS_IT_STOPI`

`SMBUS_IT_NACKI`

`SMBUS_IT_ADDRI`

`SMBUS_IT_RXI`

SMBUS\_IT\_TXI

SMBUS\_IT\_TX

SMBUS\_IT\_RX

SMBUS\_IT\_ALERT

SMBUS\_IT\_ADDR

***SMBUS nostretch mode***

SMBUS\_NOSTRETCH\_DISABLE

SMBUS\_NOSTRETCH\_ENABLE

***SMBUS ownaddress2 masks***

SMBUS\_OA2\_NOMASK

SMBUS\_OA2\_MASK01

SMBUS\_OA2\_MASK02

SMBUS\_OA2\_MASK03

SMBUS\_OA2\_MASK04

SMBUS\_OA2\_MASK05

SMBUS\_OA2\_MASK06

SMBUS\_OA2\_MASK07

***SMBUS packet error check mode***

SMBUS\_PEC\_DISABLE

SMBUS\_PEC\_ENABLE

***SMBUS peripheral mode***

SMBUS\_PERIPHERAL\_MODE\_SMBUS\_HOST

SMBUS\_PERIPHERAL\_MODE\_SMBUS\_SLAVE

SMBUS\_PERIPHERAL\_MODE\_SMBUS\_SLAVE\_ARP

***SMBUS ReloadEndMode definition***

SMBUS\_SOFTEND\_MODE

SMBUS\_RELOAD\_MODE

SMBUS\_AUTOEND\_MODE

SMBUS\_SENDPEC\_MODE

***SMBUS StartStopMode definition***

SMBUS\_NO\_STARTSTOP

SMBUS\_GENERATE\_STOP

SMBUS\_GENERATE\_START\_READ

SMBUS\_GENERATE\_START\_WRITE

***SMBUS XferOptions definition***

SMBUS\_FIRST\_FRAME  
SMBUS\_NEXT\_FRAME  
SMBUS\_FIRST\_AND\_LAST\_FRAME\_NO\_PEC  
SMBUS\_LAST\_FRAME\_NO\_PEC  
SMBUS\_FIRST\_AND\_LAST\_FRAME\_WITH\_PEC  
SMBUS\_LAST\_FRAME\_WITH\_PEC  
SMBUS\_OTHER\_FRAME\_NO\_PEC  
SMBUS\_OTHER\_FRAME\_WITH\_PEC  
SMBUS\_OTHER\_AND\_LAST\_FRAME\_NO\_PEC  
SMBUS\_OTHER\_AND\_LAST\_FRAME\_WITH\_PEC

## 48 HAL SPI Generic Driver

### 48.1 SPI Firmware driver registers structures

#### 48.1.1 SPI\_InitTypeDef

##### Data Fields

- *uint32\_t Mode*
- *uint32\_t Direction*
- *uint32\_t DataSize*
- *uint32\_t CLKPolarity*
- *uint32\_t CLKPhase*
- *uint32\_t NSS*
- *uint32\_t BaudRatePrescaler*
- *uint32\_t FirstBit*
- *uint32\_t TIMode*
- *uint32\_t CRCCalculation*
- *uint32\_t CRCPolynomial*
- *uint32\_t CRCLength*
- *uint32\_t NSSPMode*

##### Field Documentation

- ***uint32\_t SPI\_InitTypeDef::Mode***  
Specifies the SPI operating mode. This parameter can be a value of [\*\*SPI\\_Mode\*\*](#)
- ***uint32\_t SPI\_InitTypeDef::Direction***  
Specifies the SPI bidirectional mode state. This parameter can be a value of [\*\*SPI\\_Direction\*\*](#)
- ***uint32\_t SPI\_InitTypeDef::DataSize***  
Specifies the SPI data size. This parameter can be a value of [\*\*SPI\\_Data\\_Size\*\*](#)
- ***uint32\_t SPI\_InitTypeDef::CLKPolarity***  
Specifies the serial clock steady state. This parameter can be a value of [\*\*SPI\\_Clock\\_Polarity\*\*](#)
- ***uint32\_t SPI\_InitTypeDef::CLKPhase***  
Specifies the clock active edge for the bit capture. This parameter can be a value of [\*\*SPI\\_Clock\\_Phase\*\*](#)
- ***uint32\_t SPI\_InitTypeDef::NSS***  
Specifies whether the NSS signal is managed by hardware (NSS pin) or by software using the SSI bit. This parameter can be a value of [\*\*SPI\\_Slave\\_Select\\_management\*\*](#)
- ***uint32\_t SPI\_InitTypeDef::BaudRatePrescaler***  
Specifies the Baud Rate prescaler value which will be used to configure the transmit and receive SCK clock. This parameter can be a value of [\*\*SPI\\_BaudRate\\_Prescaler\*\*](#)  
**Note:** The communication clock is derived from the master clock. The slave clock does not need to be set.
- ***uint32\_t SPI\_InitTypeDef::FirstBit***  
Specifies whether data transfers start from MSB or LSB bit. This parameter can be a value of [\*\*SPI\\_MSB\\_LSB\\_transmission\*\*](#)
- ***uint32\_t SPI\_InitTypeDef::TIMode***  
Specifies if the TI mode is enabled or not. This parameter can be a value of [\*\*SPI\\_TI\\_mode\*\*](#)

- ***uint32\_t SPI\_InitTypeDef::CRCCalculation***  
Specifies if the CRC calculation is enabled or not. This parameter can be a value of **SPI\_CRC\_Calculation**
- ***uint32\_t SPI\_InitTypeDef::CRCPolynomial***  
Specifies the polynomial used for the CRC calculation. This parameter must be an odd number between Min\_Data = 1 and Max\_Data = 65535
- ***uint32\_t SPI\_InitTypeDef::CRCLength***  
Specifies the CRC Length used for the CRC calculation. CRC Length is only used with Data8 and Data16, not other data size This parameter can be a value of **SPI\_CRC\_length**
- ***uint32\_t SPI\_InitTypeDef::NSSPMode***  
Specifies whether the NSSP signal is enabled or not . This parameter can be a value of **SPI\_NSSP\_Mode** This mode is activated by the NSSP bit in the SPIx\_CR2 register and it takes effect only if the SPI interface is configured as Motorola SPI master (FRF=0) with capture on the first edge (SPIx\_CR1 CPHA = 0, CPOL setting is ignored)..

#### 48.1.2 ***\_SPI\_HandleTypeDef***

##### Data Fields

- ***SPI\_TypeDef \* Instance***
- ***SPI\_InitTypeDef Init***
- ***uint8\_t \* pTxBuffPtr***
- ***uint16\_t TxXferSize***
- ***\_\_IO uint16\_t TxXferCount***
- ***uint8\_t \* pRxBuffPtr***
- ***uint16\_t RxXferSize***
- ***\_\_IO uint16\_t RxXferCount***
- ***uint32\_t CRCSize***
- ***void(\* RxISR***
- ***void(\* TxISR***
- ***DMA\_HandleTypeDef \* hdmatx***
- ***DMA\_HandleTypeDef \* hdmarx***
- ***HAL\_LockTypeDef Lock***
- ***\_\_IO HAL\_SPI\_StateTypeDef State***
- ***\_\_IO uint32\_t ErrorCode***

##### Field Documentation

- ***SPI\_TypeDef\* \_\_SPI\_HandleTypeDef::Instance***  
SPI registers base address
- ***SPI\_InitTypeDef \_\_SPI\_HandleTypeDef::Init***  
SPI communication parameters
- ***uint8\_t\* \_\_SPI\_HandleTypeDef::pTxBuffPtr***  
Pointer to SPI Tx transfer Buffer
- ***uint16\_t \_\_SPI\_HandleTypeDef::TxXferSize***  
SPI Tx Transfer size
- ***\_\_IO uint16\_t \_\_SPI\_HandleTypeDef::TxXferCount***  
SPI Tx Transfer Counter
- ***uint8\_t\* \_\_SPI\_HandleTypeDef::pRxBuffPtr***  
Pointer to SPI Rx transfer Buffer
- ***uint16\_t \_\_SPI\_HandleTypeDef::RxXferSize***  
SPI Rx Transfer size
- ***\_\_IO uint16\_t \_\_SPI\_HandleTypeDef::RxXferCount***  
SPI Rx Transfer Counter

- **`uint32_t __SPI_HandleTypeDef::CRCSIZE`**  
SPI CRC size used for the transfer
- **`void(* __SPI_HandleTypeDef::RxISR)(struct __SPI_HandleTypeDef *hspi)`**  
function pointer on Rx ISR
- **`void(* __SPI_HandleTypeDef::TxISR)(struct __SPI_HandleTypeDef *hspi)`**  
function pointer on Tx ISR
- **`DMA_HandleTypeDef* __SPI_HandleTypeDef::hdmatx`**  
SPI Tx DMA Handle parameters
- **`DMA_HandleTypeDef* __SPI_HandleTypeDef::hdmarx`**  
SPI Rx DMA Handle parameters
- **`HAL_LockTypeDef __SPI_HandleTypeDef::Lock`**  
Locking object
- **`_IO HAL_SPI_StateTypeDef __SPI_HandleTypeDef::State`**  
SPI communication state
- **`_IO uint32_t __SPI_HandleTypeDef::ErrorCode`**  
SPI Error code

## 48.2 SPI Firmware driver API description

### 48.2.1 How to use this driver

The SPI HAL driver can be used as follows:

1. Declare a SPI\_HandleTypeDef handle structure, for example: SPI\_HandleTypeDef hspi;
2. Initialize the SPI low level resources by implementing the HAL\_SPI\_MspInit() API:
  - a. Enable the SPIx interface clock
  - b. SPI pins configuration
    - Enable the clock for the SPI GPIOs
    - Configure these SPI pins as alternate function push-pull
  - c. NVIC configuration if you need to use interrupt process
    - Configure the SPIx interrupt priority
    - Enable the NVIC SPI IRQ handle
  - d. DMA Configuration if you need to use DMA process
    - Declare a DMA\_HandleTypeDef handle structure for the transmit or receive Stream/Channel
    - Enable the DMAx clock
    - Configure the DMA handle parameters
    - Configure the DMA Tx or Rx Stream/Channel
    - Associate the initialized hdma\_tx handle to the hspi DMA Tx or Rx handle
    - Configure the priority and enable the NVIC for the transfer complete interrupt on the DMA Tx or Rx Stream/Channel
3. Program the Mode, BidirectionalMode , Data size, Baudrate Prescaler, NSS management, Clock polarity and phase, FirstBit and CRC configuration in the hspi Init structure.
4. Initialize the SPI registers by calling the HAL\_SPI\_Init() API:
  - This API configures also the low level Hardware GPIO, CLOCK, CORTEX...etc) by calling the customized HAL\_SPI\_MspInit() API.

Circular mode restriction:

1. The DMA circular mode cannot be used when the SPI is configured in these modes:
  - a. Master 2Lines RxOnly
  - b. Master 1Line Rx
2. The CRC feature is not managed when the DMA circular mode is enabled

3. When the SPI DMA Pause/Stop features are used, we must use the following APIs the HAL\_SPI\_DMAPause() / HAL\_SPI\_DMAStop() only under the SPI callbacks

Master Receive mode restriction:

1. In Master unidirectional receive-only mode (MSTR =1, BIDIMODE=0, RXONLY=0) or bidirectional receive mode (MSTR=1, BIDIMODE=1, BIDIOE=0), to ensure that the SPI does not initiate a new transfer the following procedure has to be respected:
  - a. HAL\_SPI\_DeInit()
  - b. HAL\_SPI\_Init()

The HAL drivers do not allow reaching all supported SPI frequencies in the different SPI modes. Refer to the source code (stm32xxxx\_hal\_spi.c header) to get a summary of the maximum SPI frequency that can be reached with a data size of 8 or 16 bits, depending on the APBx peripheral clock frequency (fPCLK) used by the SPI instance.

## 48.2.2 Initialization and de-initialization functions

This subsection provides a set of functions allowing to initialize and de-initialize the SPIx peripheral:

- User must implement HAL\_SPI\_MspInit() function in which he configures all related peripherals resources (CLOCK, GPIO, DMA, IT and NVIC ).
- Call the function HAL\_SPI\_Init() to configure the selected device with the selected configuration:
  - Mode
  - Direction
  - Data Size
  - Clock Polarity and Phase
  - NSS Management
  - BaudRate Prescaler
  - FirstBit
  - TIMode
  - CRC Calculation
  - CRC Polynomial if CRC enabled
  - CRC Length, used only with Data8 and Data16
  - FIFO reception threshold
- Call the function HAL\_SPI\_DeInit() to restore the default configuration of the selected SPIx peripheral.

This section contains the following APIs:

- [\*\*HAL\\_SPI\\_Init\(\)\*\*](#)
- [\*\*HAL\\_SPI\\_DeInit\(\)\*\*](#)
- [\*\*HAL\\_SPI\\_MspInit\(\)\*\*](#)
- [\*\*HAL\\_SPI\\_MspDeInit\(\)\*\*](#)

## 48.2.3 IO operation functions

This subsection provides a set of functions allowing to manage the SPI data transfers.

The SPI supports master and slave mode :

1. There are two modes of transfer:
  - Blocking mode: The communication is performed in polling mode. The HAL status of all data processing is returned by the same function after finishing transfer.
  - No-Blocking mode: The communication is performed using Interrupts or DMA, These APIs return the HAL status. The end of the data processing will be

indicated through the dedicated SPI IRQ when using Interrupt mode or the DMA IRQ when using DMA mode. The HAL\_SPI\_TxCpltCallback(), HAL\_SPI\_RxCpltCallback() and HAL\_SPI\_TxRxCpltCallback() user callbacks will be executed respectively at the end of the transmit or Receive process. The HAL\_SPI\_ErrorCallback() user callback will be executed when a communication error is detected.

2. APIs provided for these 2 transfer modes (Blocking mode or Non blocking mode using either Interrupt or DMA) exist for 1Line (simplex) and 2Lines (full duplex) modes.

This section contains the following APIs:

- [`HAL\_SPI\_Transmit\(\)`](#)
- [`HAL\_SPI\_Receive\(\)`](#)
- [`HAL\_SPI\_TransmitReceive\(\)`](#)
- [`HAL\_SPI\_Transmit\_IT\(\)`](#)
- [`HAL\_SPI\_Receive\_IT\(\)`](#)
- [`HAL\_SPI\_TransmitReceive\_IT\(\)`](#)
- [`HAL\_SPI\_Transmit\_DMA\(\)`](#)
- [`HAL\_SPI\_Receive\_DMA\(\)`](#)
- [`HAL\_SPI\_TransmitReceive\_DMA\(\)`](#)
- [`HAL\_SPI\_Abort\(\)`](#)
- [`HAL\_SPI\_Abort\_IT\(\)`](#)
- [`HAL\_SPI\_DMAPause\(\)`](#)
- [`HAL\_SPI\_DMAResume\(\)`](#)
- [`HAL\_SPI\_DMAStop\(\)`](#)
- [`HAL\_SPI\_IRQHandler\(\)`](#)
- [`HAL\_SPI\_TxCpltCallback\(\)`](#)
- [`HAL\_SPI\_RxCpltCallback\(\)`](#)
- [`HAL\_SPI\_TxRxCpltCallback\(\)`](#)
- [`HAL\_SPI\_TxHalfCpltCallback\(\)`](#)
- [`HAL\_SPI\_RxHalfCpltCallback\(\)`](#)
- [`HAL\_SPI\_TxRxHalfCpltCallback\(\)`](#)
- [`HAL\_SPI\_ErrorCallback\(\)`](#)
- [`HAL\_SPI\_AbortCpltCallback\(\)`](#)

#### 48.2.4 Peripheral State and Errors functions

This subsection provides a set of functions allowing to control the SPI.

- `HAL_SPI_GetState()` API can be helpful to check in run-time the state of the SPI peripheral
- `HAL_SPI_GetError()` check in run-time Errors occurring during communication

This section contains the following APIs:

- [`HAL\_SPI\_GetState\(\)`](#)
- [`HAL\_SPI\_GetError\(\)`](#)

#### 48.2.5 Detailed description of functions

##### **HAL\_SPI\_Init**

|                      |                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_SPI_Init (SPI_HandleTypeDef * hspi)</code>                                                             |
| Function description | Initialize the SPI according to the specified parameters in the <code>SPI_InitTypeDef</code> and initialize the associated handle. |

---

|               |                                                                                                                                                                       |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                |

### HAL\_SPI\_DelInit

|                      |                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_DelInit (SPI_HandleTypeDef * hspi)</b>                                                                                                   |
| Function description | De-Initialize the SPI peripheral.                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                |

### HAL\_SPI\_MspInit

|                      |                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SPI_MspInit (SPI_HandleTypeDef * hspi)</b>                                                                                                                |
| Function description | Initialize the SPI MSP.                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                       |

### HAL\_SPI\_MspDelInit

|                      |                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SPI_MspDelInit (SPI_HandleTypeDef * hspi)</b>                                                                                                             |
| Function description | De-Initialize the SPI MSP.                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                       |

### HAL\_SPI\_Transmit

|                      |                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_Transmit (SPI_HandleTypeDef * hspi, uint8_t * pData, uint16_t Size, uint32_t Timeout)</b>                                                                                                                                                                                              |
| Function description | Transmit an amount of data in blocking mode.                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> <li>• <b>pData:</b> pointer to data buffer</li> <li>• <b>Size:</b> amount of data to be sent</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                              |

### HAL\_SPI\_Receive

|                      |                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_Receive (SPI_HandleTypeDef * hspi, uint8_t * pData, uint16_t Size, uint32_t Timeout)</b> |
| Function description | Receive an amount of data in blocking mode.                                                                           |

|               |                                                                                                                                                                                                                                                                                                                 |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li><b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> <li><b>pData:</b> pointer to data buffer</li> <li><b>Size:</b> amount of data to be received</li> <li><b>Timeout:</b> Timeout duration</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                            |

### HAL\_SPI\_TransmitReceive

|                      |                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_TransmitReceive<br/>(SPI_HandleTypeDef * hspi, uint8_t * pTxData, uint8_t * pRxData, uint16_t Size, uint32_t Timeout)</b>                                                                                                                                                                                                                                            |
| Function description | Transmit and Receive an amount of data in blocking mode.                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> <li><b>pTxData:</b> pointer to transmission data buffer</li> <li><b>pRxData:</b> pointer to reception data buffer</li> <li><b>Size:</b> amount of data to be sent and received</li> <li><b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                              |

### HAL\_SPI\_Transmit\_IT

|                      |                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_Transmit_IT<br/>(SPI_HandleTypeDef * hspi, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                       |
| Function description | Transmit an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li><b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> <li><b>pData:</b> pointer to data buffer</li> <li><b>Size:</b> amount of data to be sent</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                              |

### HAL\_SPI\_Receive\_IT

|                      |                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_Receive_IT<br/>(SPI_HandleTypeDef * hspi, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                        |
| Function description | Receive an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li><b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> <li><b>pData:</b> pointer to data buffer</li> <li><b>Size:</b> amount of data to be sent</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                              |

**HAL\_SPI\_TransmitReceive\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_TransmitReceive_IT<br/>(SPI_HandleTypeDef * hspi, uint8_t * pTxData, uint8_t * pRxData, uint16_t Size)</b>                                                                                                                                                                                                                         |
| Function description | Transmit and Receive an amount of data in non-blocking mode with Interrupt.                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> <li>• <b>pTxData:</b> pointer to transmission data buffer</li> <li>• <b>pRxData:</b> pointer to reception data buffer</li> <li>• <b>Size:</b> amount of data to be sent and received</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                          |

**HAL\_SPI\_Transmit\_DMA**

|                      |                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_Transmit_DMA<br/>(SPI_HandleTypeDef * hspi, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                            |
| Function description | Transmit an amount of data in non-blocking mode with DMA.                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> <li>• <b>pData:</b> pointer to data buffer</li> <li>• <b>Size:</b> amount of data to be sent</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                  |

**HAL\_SPI\_Receive\_DMA**

|                      |                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_Receive_DMA<br/>(SPI_HandleTypeDef * hspi, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                             |
| Function description | Receive an amount of data in non-blocking mode with DMA.                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> <li>• <b>pData:</b> pointer to data buffer</li> <li>• <b>Size:</b> amount of data to be sent</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                  |
| Notes                | <ul style="list-style-type: none"> <li>• In case of MASTER mode and SPI_DIRECTION_2LINES direction, hdmatx shall be defined.</li> <li>• When the CRC feature is enabled the pData Length must be Size + 1.</li> </ul>                                                   |

**HAL\_SPI\_TransmitReceive\_DMA**

|                      |                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_TransmitReceive_DMA<br/>(SPI_HandleTypeDef * hspi, uint8_t * pTxData, uint8_t * pRxData, uint16_t Size)</b> |
| Function description | Transmit and Receive an amount of data in non-blocking mode with DMA.                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains</li> </ul>                  |

|               |                                                                       |
|---------------|-----------------------------------------------------------------------|
|               | the configuration information for SPI module.                         |
|               | • <b>pTxData:</b> pointer to transmission data buffer                 |
|               | • <b>pRxData:</b> pointer to reception data buffer                    |
|               | • <b>Size:</b> amount of data to be sent                              |
| Return values | • <b>HAL:</b> status                                                  |
| Notes         | • When the CRC feature is enabled the pRxData Length must be Size + 1 |

### **HAL\_SPI\_DMAPause**

|                      |                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_DMAPause (SPI_HandleTypeDef * hspi)</b>                                                              |
| Function description | Pause the DMA Transfer.                                                                                                           |
| Parameters           | • <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for the specified SPI module. |
| Return values        | • <b>HAL:</b> status                                                                                                              |

### **HAL\_SPI\_DMAResume**

|                      |                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_DMAResume (SPI_HandleTypeDef * hspi)</b>                                                             |
| Function description | Resume the DMA Transfer.                                                                                                          |
| Parameters           | • <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for the specified SPI module. |
| Return values        | • <b>HAL:</b> status                                                                                                              |

### **HAL\_SPI\_DMAStop**

|                      |                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_DMAStop (SPI_HandleTypeDef * hspi)</b>                                                               |
| Function description | Stop the DMA Transfer.                                                                                                            |
| Parameters           | • <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for the specified SPI module. |
| Return values        | • <b>HAL:</b> status                                                                                                              |

### **HAL\_SPI\_Abort**

|                      |                                                                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_Abort (SPI_HandleTypeDef * hspi)</b>                                                                                                                                                                      |
| Function description | Abort ongoing transfer (blocking mode).                                                                                                                                                                                                |
| Parameters           | • <b>hspi:</b> SPI handle.                                                                                                                                                                                                             |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                   |
| Notes                | • This procedure could be used for aborting any ongoing transfer (Tx and Rx), started in Interrupt or DMA mode. This procedure performs following operations : Disable SPI Interrupts (depending of transfer direction)Disable the DMA |

- transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL\_DMA\_Abort (in case of transfer in DMA mode)Set handle State to READY
- This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.

### **HAL\_SPI\_Abort\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SPI_Abort_IT (SPI_HandleTypeDef * hspi)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Abort ongoing transfer (Interrupt mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> SPI handle.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing transfer (Tx and Rx), started in Interrupt or DMA mode. This procedure performs following operations : Disable SPI Interrupts (depending of transfer direction)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort_IT (in case of transfer in DMA mode)Set handle State to READYAt abort completion, call user abort complete callback</li> <li>• This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).</li> </ul> |

### **HAL\_SPI\_IRQHandler**

|                      |                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SPI_IRQHandler (SPI_HandleTypeDef * hspi)</b>                                                                                                                           |
| Function description | Handle SPI interrupt request.                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for the specified SPI module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                     |

### **HAL\_SPI\_TxCpltCallback**

|                      |                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SPI_TxCpltCallback (SPI_HandleTypeDef * hspi)</b>                                                                                                         |
| Function description | Tx Transfer completed callback.                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                       |

### **HAL\_SPI\_RxCpltCallback**

|                      |                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SPI_RxCpltCallback (SPI_HandleTypeDef * hspi)</b>                                                           |
| Function description | Rx Transfer completed callback.                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains</li> </ul> |

the configuration information for SPI module.

Return values

- **None**

### **HAL\_SPI\_TxRxCpltCallback**

Function name **void HAL\_SPI\_TxRxCpltCallback (SPI\_HandleTypeDef \* hspi)**

Function description Tx and Rx Transfer completed callback.

Parameters

- **hspi:** pointer to a SPI\_HandleTypeDef structure that contains the configuration information for SPI module.

Return values

- **None**

### **HAL\_SPI\_TxHalfCpltCallback**

Function name **void HAL\_SPI\_TxHalfCpltCallback (SPI\_HandleTypeDef \* hspi)**

Function description Tx Half Transfer completed callback.

Parameters

- **hspi:** pointer to a SPI\_HandleTypeDef structure that contains the configuration information for SPI module.

Return values

- **None**

### **HAL\_SPI\_RxHalfCpltCallback**

Function name **void HAL\_SPI\_RxHalfCpltCallback (SPI\_HandleTypeDef \* hspi)**

Function description Rx Half Transfer completed callback.

Parameters

- **hspi:** pointer to a SPI\_HandleTypeDef structure that contains the configuration information for SPI module.

Return values

- **None**

### **HAL\_SPI\_TxRxHalfCpltCallback**

Function name **void HAL\_SPI\_TxRxHalfCpltCallback (SPI\_HandleTypeDef \* hspi)**

Function description Tx and Rx Half Transfer callback.

Parameters

- **hspi:** pointer to a SPI\_HandleTypeDef structure that contains the configuration information for SPI module.

Return values

- **None**

### **HAL\_SPI\_ErrorCallback**

Function name **void HAL\_SPI\_ErrorCallback (SPI\_HandleTypeDef \* hspi)**

Function description SPI error callback.

Parameters

- **hspi:** pointer to a SPI\_HandleTypeDef structure that contains the configuration information for SPI module.

Return values

- **None**

**HAL\_SPI\_AbortCpltCallback**

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| Function name        | <b>void HAL_SPI_AbortCpltCallback (SPI_HandleTypeDef * hspi)</b>           |
| Function description | SPI Abort Complete callback.                                               |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hspi:</b> SPI handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>              |

**HAL\_SPI\_GetState**

|                      |                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_SPI_StateTypeDef HAL_SPI_GetState (SPI_HandleTypeDef * hspi)</b>                                                                                             |
| Function description | Return the SPI handle state.                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>SPI:</b> state</li></ul>                                                                                                 |

**HAL\_SPI\_GetError**

|                      |                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_SPI_GetError (SPI_HandleTypeDef * hspi)</b>                                                                                                         |
| Function description | Return the SPI error code.                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hspi:</b> pointer to a SPI_HandleTypeDef structure that contains the configuration information for SPI module.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>SPI:</b> error code in bitmap format</li></ul>                                                                           |

## 48.3 SPI Firmware driver defines

### 48.3.1 SPI

***SPI BaudRate Prescaler***

SPI\_BAUDRATEPRESCALER\_2  
SPI\_BAUDRATEPRESCALER\_4  
SPI\_BAUDRATEPRESCALER\_8  
SPI\_BAUDRATEPRESCALER\_16  
SPI\_BAUDRATEPRESCALER\_32  
SPI\_BAUDRATEPRESCALER\_64  
SPI\_BAUDRATEPRESCALER\_128  
SPI\_BAUDRATEPRESCALER\_256

***SPI Clock Phase***

SPI\_PHASE\_1EDGE  
SPI\_PHASE\_2EDGE

***SPI Clock Polarity***

SPI\_POLARITY\_LOW

SPI\_POLARITY\_HIGH

***SPI CRC Calculation***

SPI\_CRCALCULATION\_DISABLE

SPI\_CRCALCULATION\_ENABLE

***SPI CRC Length***

SPI\_CRC\_LENGTH\_DATASIZE

SPI\_CRC\_LENGTH\_8BIT

SPI\_CRC\_LENGTH\_16BIT

***SPI Data Size***

SPI\_DATASIZE\_4BIT

SPI\_DATASIZE\_5BIT

SPI\_DATASIZE\_6BIT

SPI\_DATASIZE\_7BIT

SPI\_DATASIZE\_8BIT

SPI\_DATASIZE\_9BIT

SPI\_DATASIZE\_10BIT

SPI\_DATASIZE\_11BIT

SPI\_DATASIZE\_12BIT

SPI\_DATASIZE\_13BIT

SPI\_DATASIZE\_14BIT

SPI\_DATASIZE\_15BIT

SPI\_DATASIZE\_16BIT

***SPI Direction Mode***

SPI\_DIRECTION\_2LINES

SPI\_DIRECTION\_2LINES\_RXONLY

SPI\_DIRECTION\_1LINE

***SPI Error Code***

HAL\_SPI\_ERROR\_NONE No error

HAL\_SPI\_ERROR\_MODF MODF error

HAL\_SPI\_ERROR\_CRC CRC error

HAL\_SPI\_ERROR\_OVR OVR error

HAL\_SPI\_ERROR\_FRE FRE error

HAL\_SPI\_ERROR\_DMA DMA transfer error

HAL\_SPI\_ERROR\_FLAG Error on RXNE/TXE/BSY/FTLVL/FRLVL Flag

HAL\_SPI\_ERROR\_ABORT Error during SPI Abort procedure

**SPI Exported Macros**`__HAL_SPI_RESET_HANDLE_STATE`**Description:**

- Reset SPI handle state.

**Parameters:**

- `__HANDLE__`: specifies the SPI Handle.  
This parameter can be SPI where x: 1, 2, or 3 to select the SPI peripheral.

**Return value:**

- None

`__HAL_SPI_ENABLE_IT`**Description:**

- Enable the specified SPI interrupts.

**Parameters:**

- `__HANDLE__`: specifies the SPI Handle.  
This parameter can be SPI where x: 1, 2, or 3 to select the SPI peripheral.
- `__INTERRUPT__`: specifies the interrupt source to enable. This parameter can be one of the following values:
  - `SPI_IT_TXE`: Tx buffer empty interrupt enable
  - `SPI_IT_RXNE`: RX buffer not empty interrupt enable
  - `SPI_IT_ERR`: Error interrupt enable

**Return value:**

- None

`__HAL_SPI_DISABLE_IT`**Description:**

- Disable the specified SPI interrupts.

**Parameters:**

- `__HANDLE__`: specifies the SPI handle.  
This parameter can be SPIx where x: 1, 2, or 3 to select the SPI peripheral.
- `__INTERRUPT__`: specifies the interrupt source to disable. This parameter can be one of the following values:
  - `SPI_IT_TXE`: Tx buffer empty interrupt enable
  - `SPI_IT_RXNE`: RX buffer not empty interrupt enable
  - `SPI_IT_ERR`: Error interrupt enable

**Return value:**

- None

`__HAL_SPI_GET_IT_SOURCE`**Description:**

- Check whether the specified SPI interrupt

source is enabled or not.

**Parameters:**

- HANDLE: specifies the SPI Handle. This parameter can be SPI where x: 1, 2, or 3 to select the SPI peripheral.
- INTERRUPT: specifies the SPI interrupt source to check. This parameter can be one of the following values:
  - SPI\_IT\_TXE: Tx buffer empty interrupt enable
  - SPI\_IT\_RXNE: RX buffer not empty interrupt enable
  - SPI\_IT\_ERR: Error interrupt enable

**Return value:**

- The: new state of IT (TRUE or FALSE).

\_HAL\_SPI\_GET\_FLAG

- Description:**
- Check whether the specified SPI flag is set or not.

**Parameters:**

- HANDLE: specifies the SPI Handle. This parameter can be SPI where x: 1, 2, or 3 to select the SPI peripheral.
- FLAG: specifies the flag to check. This parameter can be one of the following values:
  - SPI\_FLAG\_RXNE: Receive buffer not empty flag
  - SPI\_FLAG\_TXE: Transmit buffer empty flag
  - SPI\_FLAG\_CRCERR: CRC error flag
  - SPI\_FLAG\_MODF: Mode fault flag
  - SPI\_FLAG\_OVR: Overrun flag
  - SPI\_FLAG\_BSY: Busy flag
  - SPI\_FLAG\_FRE: Frame format error flag
  - SPI\_FLAG\_FTLVL: SPI fifo transmission level
  - SPI\_FLAG\_FRLVL: SPI fifo reception level

**Return value:**

- The: new state of FLAG (TRUE or FALSE).

\_HAL\_SPI\_CLEAR\_CRCERRFLAG

**Description:**

- Clear the SPI CRCERR pending flag.

**Parameters:**

- HANDLE: specifies the SPI Handle. This parameter can be SPI where x: 1, 2, or 3 to select the SPI peripheral.

3 to select the SPI peripheral.

**Return value:**

- None

`__HAL_SPI_CLEAR_MODFLAG`

**Description:**

- Clear the SPI MODF pending flag.

**Parameters:**

- `__HANDLE__`: specifies the SPI Handle.  
This parameter can be SPI where x: 1, 2, or  
3 to select the SPI peripheral.

**Return value:**

- None

`__HAL_SPI_CLEAR_OVRFLAG`

**Description:**

- Clear the SPI OVR pending flag.

**Parameters:**

- `__HANDLE__`: specifies the SPI Handle.  
This parameter can be SPI where x: 1, 2, or  
3 to select the SPI peripheral.

**Return value:**

- None

`__HAL_SPI_CLEAR_FREFLAG`

**Description:**

- Clear the SPI FRE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the SPI Handle.  
This parameter can be SPI where x: 1, 2, or  
3 to select the SPI peripheral.

**Return value:**

- None

`__HAL_SPI_ENABLE`

**Description:**

- Enable the SPI peripheral.

**Parameters:**

- `__HANDLE__`: specifies the SPI Handle.  
This parameter can be SPI where x: 1, 2, or  
3 to select the SPI peripheral.

**Return value:**

- None

`__HAL_SPI_DISABLE`

**Description:**

- Disable the SPI peripheral.

**Parameters:**

- `__HANDLE__`: specifies the SPI Handle.

This parameter can be SPI where x: 1, 2, or 3 to select the SPI peripheral.

**Return value:**

- None

***SPI FIFO Reception Threshold***

SPI\_RXFIFO\_THRESHOLD  
SPI\_RXFIFO\_THRESHOLD\_QF  
SPI\_RXFIFO\_THRESHOLD\_HF

***SPI Flags Definition***

SPI\_FLAG\_RXNE  
SPI\_FLAG\_TXE  
SPI\_FLAG\_BSY  
SPI\_FLAG\_CRCERR  
SPI\_FLAG\_MODF  
SPI\_FLAG\_OVR  
SPI\_FLAG\_FRE  
SPI\_FLAG\_FTLVL  
SPI\_FLAG\_FRLVL

***SPI Interrupt Definition***

SPI\_IT\_TXE  
SPI\_IT\_RXNE  
SPI\_IT\_ERR

***SPI Mode***

SPI\_MODE\_SLAVE  
SPI\_MODE\_MASTER

***SPI MSB LSB Transmission***

SPI\_FIRSTBIT\_MSB  
SPI\_FIRSTBIT\_LSB

***SPI NSS Pulse Mode***

SPI\_NSS\_PULSE\_ENABLE  
SPI\_NSS\_PULSE\_DISABLE

***SPI Reception FIFO Status Level***

SPI\_FRLVL\_EMPTY  
SPI\_FRLVL\_QUARTER\_FULL  
SPI\_FRLVL\_HALF\_FULL  
SPI\_FRLVL\_FULL

***SPI Slave Select Management***

SPI\_NSS\_SOFT

SPI\_NSS\_HARD\_INPUT

SPI\_NSS\_HARD\_OUTPUT

***SPI TI Mode***

SPI\_TIMODE\_DISABLE

SPI\_TIMODE\_ENABLE

***SPI Transmission FIFO Status Level***

SPI\_FTLVL\_EMPTY

SPI\_FTLVL\_QUARTER\_FULL

SPI\_FTLVL\_HALF\_FULL

SPI\_FTLVL\_FULL

## 49 HAL SPI Extension Driver

### 49.1 SPIEx Firmware driver API description

#### 49.1.1 IO operation functions

This subsection provides a set of extended functions to manage the SPI data transfers.

1. Rx data flush function:
  - HAL\_SPIEx\_FlushRxFifo()

This section contains the following APIs:

- [\*HAL\\_SPIEx\\_FlushRxFifo\(\)\*](#)

#### 49.1.2 Detailed description of functions

##### **HAL\_SPIEx\_FlushRxFifo**

Function name            **HAL\_StatusTypeDef HAL\_SPIEx\_FlushRxFifo  
(SPI\_HandleTypeDef \* hspi)**

Function description    Flush the RX fifo.

Parameters              • **hspi**: pointer to a SPI\_HandleTypeDef structure that contains the configuration information for the specified SPI module.

Return values            • **HAL**: status

## 50 HAL SRAM Generic Driver

### 50.1 SRAM Firmware driver registers structures

#### 50.1.1 SRAM\_HandleTypeDef

##### Data Fields

- *FMC\_NORSRAM\_TypeDef \* Instance*
- *FMC\_NORSRAM\_EXTENDED\_TypeDef \* Extended*
- *FMC\_NORSRAM\_InitTypeDef Init*
- *HAL\_LockTypeDef Lock*
- *\_\_IO HAL\_SRAM\_StateTypeDef State*
- *DMA\_HandleTypeDef \* hdma*

##### Field Documentation

- ***FMC\_NORSRAM\_TypeDef\* SRAM\_HandleTypeDef::Instance***  
Register base address
- ***FMC\_NORSRAM\_EXTENDED\_TypeDef\* SRAM\_HandleTypeDef::Extended***  
Extended mode register base address
- ***FMC\_NORSRAM\_InitTypeDef SRAM\_HandleTypeDef::Init***  
SRAM device control configuration parameters
- ***HAL\_LockTypeDef SRAM\_HandleTypeDef::Lock***  
SRAM locking object
- ***\_\_IO HAL\_SRAM\_StateTypeDef SRAM\_HandleTypeDef::State***  
SRAM device access state
- ***DMA\_HandleTypeDef\* SRAM\_HandleTypeDef::hdma***  
Pointer DMA handler

### 50.2 SRAM Firmware driver API description

#### 50.2.1 How to use this driver

This driver is a generic layered driver which contains a set of APIs used to control SRAM memories. It uses the FMC layer functions to interface with SRAM devices. The following sequence should be followed to configure the FMC to interface with SRAM/PSRAM memories:

1. Declare a SRAM\_HandleTypeDef handle structure, for example:  
SRAM\_HandleTypeDef hsram; and:
  - Fill the SRAM\_HandleTypeDef handle "Init" field with the allowed values of the structure member.
  - Fill the SRAM\_HandleTypeDef handle "Instance" field with a predefined base register instance for NOR or SRAM device
  - Fill the SRAM\_HandleTypeDef handle "Extended" field with a predefined base register instance for NOR or SRAM extended mode
2. Declare two FMC\_NORSRAM\_TimingTypeDef structures, for both normal and extended mode timings; for example: FMC\_NORSRAM\_TimingTypeDef Timing and FMC\_NORSRAM\_TimingTypeDef ExTiming; and fill its fields with the allowed values of the structure member.
3. Initialize the SRAM Controller by calling the function HAL\_SRAM\_Init(). This function performs the following sequence:
  - a. MSP hardware layer configuration using the function HAL\_SRAM\_MspInit()

- b. Control register configuration using the FMC NORSRAM interface function `FMC_NORSRAM_Init()`
  - c. Timing register configuration using the FMC NORSRAM interface function `FMC_NORSRAM_Timing_Init()`
  - d. Extended mode Timing register configuration using the FMC NORSRAM interface function `FMC_NORSRAM_Extended_Timing_Init()`
  - e. Enable the SRAM device using the macro `__FMC_NORSRAM_ENABLE()`
4. At this stage you can perform read/write accesses from/to the memory connected to the NOR/SRAM Bank. You can perform either polling or DMA transfer using the following APIs:
    - `HAL_SRAM_Read()`/`HAL_SRAM_Write()` for polling read/write access
    - `HAL_SRAM_Read_DMA()`/`HAL_SRAM_Write_DMA()` for DMA read/write transfer
  5. You can also control the SRAM device by calling the control APIs `HAL_SRAM_WriteOperation_Enable()`/`HAL_SRAM_WriteOperation_Disable()` to respectively enable/disable the SRAM write operation
  6. You can continuously monitor the SRAM device HAL state by calling the function `HAL_SRAM_GetState()`

### 50.2.2 SRAM Initialization and de\_initialization functions

This section provides functions allowing to initialize/de-initialize the SRAM memory

This section contains the following APIs:

- `HAL_SRAM_Init()`
- `HAL_SRAM_DeInit()`
- `HAL_SRAM_MspInit()`
- `HAL_SRAM_MspDeInit()`
- `HAL_SRAM_DMA_XferCpltCallback()`
- `HAL_SRAM_DMA_XferErrorCallback()`

### 50.2.3 SRAM Input and Output functions

This section provides functions allowing to use and control the SRAM memory

This section contains the following APIs:

- `HAL_SRAM_Read_8b()`
- `HAL_SRAM_Write_8b()`
- `HAL_SRAM_Read_16b()`
- `HAL_SRAM_Write_16b()`
- `HAL_SRAM_Read_32b()`
- `HAL_SRAM_Write_32b()`
- `HAL_SRAM_Read_DMA()`
- `HAL_SRAM_Write_DMA()`

### 50.2.4 SRAM Control functions

This subsection provides a set of functions allowing to control dynamically the SRAM interface.

This section contains the following APIs:

- `HAL_SRAM_WriteOperation_Enable()`
- `HAL_SRAM_WriteOperation_Disable()`

## 50.2.5 SRAM State functions

This subsection permits to get in run-time the status of the SRAM controller and the data flow.

This section contains the following APIs:

- [\*\*HAL\\_SRAM\\_GetState\(\)\*\*](#)

## 50.2.6 Detailed description of functions

### **HAL\_SRAM\_Init**

|                      |                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_Init (SRAM_HandleTypeDef * hsram, FMC_NORSRAM_TimingTypeDef * Timing, FMC_NORSRAM_TimingTypeDef * ExtTiming)</b>                                                                                                                                                                          |
| Function description | Performs the SRAM device initialization sequence.                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> <li>• <b>Timing:</b> Pointer to SRAM control timing structure</li> <li>• <b>ExtTiming:</b> Pointer to SRAM extended mode timing structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                  |

### **HAL\_SRAM\_DelInit**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_DelInit (SRAM_HandleTypeDef * hsram)</b>                                                                                                   |
| Function description | Performs the SRAM device De-initialization sequence.                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                   |

### **HAL\_SRAM\_MspInit**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SRAM_MspInit (SRAM_HandleTypeDef * hsram)</b>                                                                                                                |
| Function description | SRAM MSP Init.                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                          |

### **HAL\_SRAM\_MspDelInit**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SRAM_MspDelInit (SRAM_HandleTypeDef * hsram)</b>                                                                                                             |
| Function description | SRAM MSP DelInit.                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                          |

**HAL\_SRAM\_DMA\_XferCpltCallback**

|                      |                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SRAM_DMA_XferCpltCallback<br/>(DMA_HandleTypeDef * hdma)</b>                                                                                                |
| Function description | DMA transfer complete callback.                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                         |

**HAL\_SRAM\_DMA\_XferErrorCallback**

|                      |                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_SRAM_DMA_XferErrorCallback<br/>(DMA_HandleTypeDef * hdma)</b>                                                                                               |
| Function description | DMA transfer complete error callback.                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                         |

**HAL\_SRAM\_Read\_8b**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_Read_8b<br/>(SRAM_HandleTypeDef * hsram, uint32_t * pAddress, uint8_t * pDstBuffer, uint32_t BufferSize)</b>                                                                                                                                                                                                                          |
| Function description | Reads 8-bit buffer from SRAM memory.                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> <li>• <b>pAddress:</b> Pointer to read start address</li> <li>• <b>pDstBuffer:</b> Pointer to destination buffer</li> <li>• <b>BufferSize:</b> Size of the buffer to read from memory</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                              |

**HAL\_SRAM\_Write\_8b**

|                      |                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_Write_8b<br/>(SRAM_HandleTypeDef * hsram, uint32_t * pAddress, uint8_t * pSrcBuffer, uint32_t BufferSize)</b>                                                                                                                                                                                                                             |
| Function description | Writes 8-bit buffer to SRAM memory.                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> <li>• <b>pAddress:</b> Pointer to write start address</li> <li>• <b>pSrcBuffer:</b> Pointer to source buffer to write</li> <li>• <b>BufferSize:</b> Size of the buffer to write to memory</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                  |

**HAL\_SRAM\_Read\_16b**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_Read_16b<br/>(SRAM_HandleTypeDef * hsram, uint32_t * pAddress, uint16_t<br/>* pDstBuffer, uint32_t BufferSize)</b>                                                                                                                                                                                                                    |
| Function description | Reads 16-bit buffer from SRAM memory.                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> <li>• <b>pAddress:</b> Pointer to read start address</li> <li>• <b>pDstBuffer:</b> Pointer to destination buffer</li> <li>• <b>BufferSize:</b> Size of the buffer to read from memory</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                              |

**HAL\_SRAM\_Write\_16b**

|                      |                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_Write_16b<br/>(SRAM_HandleTypeDef * hsram, uint32_t * pAddress, uint16_t<br/>* pSrcBuffer, uint32_t BufferSize)</b>                                                                                                                                                                                                                       |
| Function description | Writes 16-bit buffer to SRAM memory.                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> <li>• <b>pAddress:</b> Pointer to write start address</li> <li>• <b>pSrcBuffer:</b> Pointer to source buffer to write</li> <li>• <b>BufferSize:</b> Size of the buffer to write to memory</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                  |

**HAL\_SRAM\_Read\_32b**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_Read_32b<br/>(SRAM_HandleTypeDef * hsram, uint32_t * pAddress, uint32_t<br/>* pDstBuffer, uint32_t BufferSize)</b>                                                                                                                                                                                                                    |
| Function description | Reads 32-bit buffer from SRAM memory.                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> <li>• <b>pAddress:</b> Pointer to read start address</li> <li>• <b>pDstBuffer:</b> Pointer to destination buffer</li> <li>• <b>BufferSize:</b> Size of the buffer to read from memory</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                              |

**HAL\_SRAM\_Write\_32b**

|                      |                                                                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_Write_32b<br/>(SRAM_HandleTypeDef * hsram, uint32_t * pAddress, uint32_t<br/>* pSrcBuffer, uint32_t BufferSize)</b>                                                                                                                                                   |
| Function description | Writes 32-bit buffer to SRAM memory.                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> <li>• <b>pAddress:</b> Pointer to write start address</li> <li>• <b>pSrcBuffer:</b> Pointer to source buffer to write</li> </ul> |

- **BufferSize:** Size of the buffer to write to memory
- **HAL:** status

### **HAL\_SRAM\_Read\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_Read_DMA<br/>(SRAM_HandleTypeDef * hsram, uint32_t * pAddress, uint32_t<br/>* pDstBuffer, uint32_t BufferSize)</b>                                                                                                                                                                                                                    |
| Function description | Reads a Words data from the SRAM memory using DMA transfer.                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> <li>• <b>pAddress:</b> Pointer to read start address</li> <li>• <b>pDstBuffer:</b> Pointer to destination buffer</li> <li>• <b>BufferSize:</b> Size of the buffer to read from memory</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                |

### **HAL\_SRAM\_Write\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_Write_DMA<br/>(SRAM_HandleTypeDef * hsram, uint32_t * pAddress, uint32_t<br/>* pSrcBuffer, uint32_t BufferSize)</b>                                                                                                                                                                                                                       |
| Function description | Writes a Words data buffer to SRAM memory using DMA transfer.                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> <li>• <b>pAddress:</b> Pointer to write start address</li> <li>• <b>pSrcBuffer:</b> Pointer to source buffer to write</li> <li>• <b>BufferSize:</b> Size of the buffer to write to memory</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                    |

### **HAL\_SRAM\_WriteOperation\_Enable**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_WriteOperation_Enable<br/>(SRAM_HandleTypeDef * hsram)</b>                                                                                 |
| Function description | Enables dynamically SRAM write operation.                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                     |

### **HAL\_SRAM\_WriteOperation\_Disable**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_SRAM_WriteOperation_Disable<br/>(SRAM_HandleTypeDef * hsram)</b>                                                                                |
| Function description | Disables dynamically SRAM write operation.                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                     |

**HAL\_SRAM\_GetState**

|                      |                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_SRAM_StateTypeDef HAL_SRAM_GetState<br/>(SRAM_HandleTypeDef * hsram)</b>                                                                                        |
| Function description | Returns the SRAM controller state.                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"><li>• <b>hsram:</b> pointer to a SRAM_HandleTypeDef structure that contains the configuration information for SRAM module.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> state</li></ul>                                                                                                    |

## 50.3 SRAM Firmware driver defines

### 50.3.1 SRAM

***SRAM Exported Macros***

**\_\_HAL\_SRAM\_RESET\_HANDLE\_STATE** **Description:**

- Reset SRAM handle state.

**Parameters:**

- **\_\_HANDLE\_\_**: SRAM handle

**Return value:**

- None

## 51 HAL TIM Generic Driver

### 51.1 TIM Firmware driver registers structures

#### 51.1.1 TIM\_Base\_InitTypeDef

##### Data Fields

- *uint32\_t Prescaler*
- *uint32\_t CounterMode*
- *uint32\_t Period*
- *uint32\_t ClockDivision*
- *uint32\_t RepetitionCounter*
- *uint32\_t AutoReloadPreload*

##### Field Documentation

- ***uint32\_t TIM\_Base\_InitTypeDef::Prescaler***  
Specifies the prescaler value used to divide the TIM clock. This parameter can be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFFU
- ***uint32\_t TIM\_Base\_InitTypeDef::CounterMode***  
Specifies the counter mode. This parameter can be a value of [\*\*TIM\\_Counter\\_Mode\*\*](#)
- ***uint32\_t TIM\_Base\_InitTypeDef::Period***  
Specifies the period value to be loaded into the active Auto-Reload Register at the next update event. This parameter can be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFF.
- ***uint32\_t TIM\_Base\_InitTypeDef::ClockDivision***  
Specifies the clock division. This parameter can be a value of [\*\*TIM\\_ClockDivision\*\*](#)
- ***uint32\_t TIM\_Base\_InitTypeDef::RepetitionCounter***  
Specifies the repetition counter value. Each time the RCR downcounter reaches zero, an update event is generated and counting restarts from the RCR value (N). This means in PWM mode that (N+1U) corresponds to:the number of PWM periods in edge-aligned mode the number of half PWM period in center-aligned mode GP timers: this parameter must be a number between Min\_Data = 0x00 and Max\_Data = 0xFF. Advanced timers: this parameter must be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFF.
- ***uint32\_t TIM\_Base\_InitTypeDef::AutoReloadPreload***  
Specifies the auto-reload preload. This parameter can be a value of [\*\*TIM\\_AutoReloadPreload\*\*](#)

#### 51.1.2 TIM\_OC\_InitTypeDef

##### Data Fields

- *uint32\_t OCMode*
- *uint32\_t Pulse*
- *uint32\_t OCPolarity*
- *uint32\_t OCNPolarity*
- *uint32\_t OCFastMode*
- *uint32\_t OCIdleState*
- *uint32\_t OCNIdleState*

### Field Documentation

- **`uint32_t TIM_OC_InitTypeDef::OCMode`**  
Specifies the TIM mode. This parameter can be a value of  
[`TIMEx\_Output\_Compare\_and\_PWM\_modes`](#)
- **`uint32_t TIM_OC_InitTypeDef::Pulse`**  
Specifies the pulse value to be loaded into the Capture Compare Register. This parameter can be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFFU
- **`uint32_t TIM_OC_InitTypeDef::OCPolarity`**  
Specifies the output polarity. This parameter can be a value of  
[`TIM\_Output\_Compare\_Polarity`](#)
- **`uint32_t TIM_OC_InitTypeDef::OCNPolarity`**  
Specifies the complementary output polarity. This parameter can be a value of  
[`TIM\_Output\_Compare\_N\_Polarity`](#)  
**Note:**This parameter is valid only for TIM1 and TIM8.
- **`uint32_t TIM_OC_InitTypeDef::OCFastMode`**  
Specifies the Fast mode state. This parameter can be a value of  
[`TIM\_Output\_Fast\_State`](#)  
**Note:**This parameter is valid only in PWM1 and PWM2 mode.
- **`uint32_t TIM_OC_InitTypeDef::OCIdleState`**  
Specifies the TIM Output Compare pin state during Idle state. This parameter can be a value of  
[`TIM\_Output\_Compare\_Idle\_State`](#)  
**Note:**This parameter is valid only for TIM1 and TIM8.
- **`uint32_t TIM_OC_InitTypeDef::OCNIdleState`**  
Specifies the TIM Output Compare pin state during Idle state. This parameter can be a value of  
[`TIM\_Output\_Compare\_N\_Idle\_State`](#)  
**Note:**This parameter is valid only for TIM1 and TIM8.

### 51.1.3 `TIM_OnePulse_InitTypeDef`

#### Data Fields

- **`uint32_t OCMode`**
- **`uint32_t Pulse`**
- **`uint32_t OCPolarity`**
- **`uint32_t OCNPolarity`**
- **`uint32_t OCIdleState`**
- **`uint32_t OCNIdleState`**
- **`uint32_t IC_Polarity`**
- **`uint32_t IC_Selection`**
- **`uint32_t IC_Filter`**

#### Field Documentation

- **`uint32_t TIM_OnePulse_InitTypeDef::OCMode`**  
Specifies the TIM mode. This parameter can be a value of  
[`TIMEx\_Output\_Compare\_and\_PWM\_modes`](#)
- **`uint32_t TIM_OnePulse_InitTypeDef::Pulse`**  
Specifies the pulse value to be loaded into the Capture Compare Register. This parameter can be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFFU
- **`uint32_t TIM_OnePulse_InitTypeDef::OCPolarity`**  
Specifies the output polarity. This parameter can be a value of  
[`TIM\_Output\_Compare\_Polarity`](#)
- **`uint32_t TIM_OnePulse_InitTypeDef::OCNPolarity`**  
Specifies the complementary output polarity. This parameter can be a value of  
[`TIM\_Output\_Compare\_N\_Polarity`](#)  
**Note:**This parameter is valid only for TIM1 and TIM8.

- ***uint32\_t TIM\_OnePulse\_InitTypeDef::OCIdleState***  
Specifies the TIM Output Compare pin state during Idle state. This parameter can be a value of [\*\*TIM\\_Output\\_Compare\\_Idle\\_State\*\*](#)  
**Note:**This parameter is valid only for TIM1 and TIM8.
- ***uint32\_t TIM\_OnePulse\_InitTypeDef::OCNIdleState***  
Specifies the TIM Output Compare pin state during Idle state. This parameter can be a value of [\*\*TIM\\_Output\\_Compare\\_N\\_Idle\\_State\*\*](#)  
**Note:**This parameter is valid only for TIM1 and TIM8.
- ***uint32\_t TIM\_OnePulse\_InitTypeDef::ICPolarity***  
Specifies the active edge of the input signal. This parameter can be a value of [\*\*TIM\\_Input\\_Capture\\_Polarity\*\*](#)
- ***uint32\_t TIM\_OnePulse\_InitTypeDef::ICSelection***  
Specifies the input. This parameter can be a value of [\*\*TIM\\_Input\\_Capture\\_Selection\*\*](#)
- ***uint32\_t TIM\_OnePulse\_InitTypeDef::ICFilter***  
Specifies the input capture filter. This parameter can be a number between Min\_Data = 0x0 and Max\_Data = 0xFU

#### 51.1.4 **TIM\_IC\_InitTypeDef**

##### Data Fields

- ***uint32\_t IC\_Polarity***
- ***uint32\_t IC\_Selection***
- ***uint32\_t IC\_Prescaler***
- ***uint32\_t IC\_Filter***

##### Field Documentation

- ***uint32\_t TIM\_IC\_InitTypeDef::ICPolarity***  
Specifies the active edge of the input signal. This parameter can be a value of [\*\*TIM\\_Input\\_Capture\\_Polarity\*\*](#)
- ***uint32\_t TIM\_IC\_InitTypeDef::ICSelection***  
Specifies the input. This parameter can be a value of [\*\*TIM\\_Input\\_Capture\\_Selection\*\*](#)
- ***uint32\_t TIM\_IC\_InitTypeDef::ICPrescaler***  
Specifies the Input Capture Prescaler. This parameter can be a value of [\*\*TIM\\_Input\\_Capture\\_Prescaler\*\*](#)
- ***uint32\_t TIM\_IC\_InitTypeDef::ICFilter***  
Specifies the input capture filter. This parameter can be a number between Min\_Data = 0x0 and Max\_Data = 0xFU

#### 51.1.5 **TIM\_Encoder\_InitTypeDef**

##### Data Fields

- ***uint32\_t EncoderMode***
- ***uint32\_t IC1Polarity***
- ***uint32\_t IC1Selection***
- ***uint32\_t IC1Prescaler***
- ***uint32\_t IC1Filter***
- ***uint32\_t IC2Polarity***
- ***uint32\_t IC2Selection***
- ***uint32\_t IC2Prescaler***
- ***uint32\_t IC2Filter***

### Field Documentation

- **`uint32_t TIM_Encoder_InitTypeDef::EncoderMode`**  
Specifies the active edge of the input signal. This parameter can be a value of [`TIM\_Encoder\_Mode`](#)
- **`uint32_t TIM_Encoder_InitTypeDef::IC1Polarity`**  
Specifies the active edge of the input signal. This parameter can be a value of [`TIM\_Input\_Capture\_Polarity`](#)
- **`uint32_t TIM_Encoder_InitTypeDef::IC1Selection`**  
Specifies the input. This parameter can be a value of [`TIM\_Input\_Capture\_Selection`](#)
- **`uint32_t TIM_Encoder_InitTypeDef::IC1Prescaler`**  
Specifies the Input Capture Prescaler. This parameter can be a value of [`TIM\_Input\_Capture\_Prescaler`](#)
- **`uint32_t TIM_Encoder_InitTypeDef::IC1Filter`**  
Specifies the input capture filter. This parameter can be a number between Min\_Data = 0x0 and Max\_Data = 0xFU
- **`uint32_t TIM_Encoder_InitTypeDef::IC2Polarity`**  
Specifies the active edge of the input signal. This parameter can be a value of [`TIM\_Input\_Capture\_Polarity`](#)
- **`uint32_t TIM_Encoder_InitTypeDef::IC2Selection`**  
Specifies the input. This parameter can be a value of [`TIM\_Input\_Capture\_Selection`](#)
- **`uint32_t TIM_Encoder_InitTypeDef::IC2Prescaler`**  
Specifies the Input Capture Prescaler. This parameter can be a value of [`TIM\_Input\_Capture\_Prescaler`](#)
- **`uint32_t TIM_Encoder_InitTypeDef::IC2Filter`**  
Specifies the input capture filter. This parameter can be a number between Min\_Data = 0x0 and Max\_Data = 0xFU

### 51.1.6 `TIM_ClockConfigTypeDef`

#### Data Fields

- **`uint32_t ClockSource`**
- **`uint32_t ClockPolarity`**
- **`uint32_t ClockPrescaler`**
- **`uint32_t ClockFilter`**

#### Field Documentation

- **`uint32_t TIM_ClockConfigTypeDef::ClockSource`**  
TIM clock sources This parameter can be a value of [`TIM\_Clock\_Source`](#)
- **`uint32_t TIM_ClockConfigTypeDef::ClockPolarity`**  
TIM clock polarity This parameter can be a value of [`TIM\_Clock\_Polarity`](#)
- **`uint32_t TIM_ClockConfigTypeDef::ClockPrescaler`**  
TIM clock prescaler This parameter can be a value of [`TIM\_Clock\_Prescaler`](#)
- **`uint32_t TIM_ClockConfigTypeDef::ClockFilter`**  
TIM clock filter This parameter can be a number between Min\_Data = 0x0 and Max\_Data = 0xFU

### 51.1.7 `TIM_ClearInputConfigTypeDef`

#### Data Fields

- **`uint32_t ClearInputState`**
- **`uint32_t ClearInputSource`**
- **`uint32_t ClearInputPolarity`**
- **`uint32_t ClearInputPrescaler`**

- *uint32\_t ClearInputFilter*

#### Field Documentation

- *uint32\_t TIM\_ClearInputConfigTypeDef::ClearInputState*  
TIM clear Input state This parameter can be ENABLE or DISABLE
- *uint32\_t TIM\_ClearInputConfigTypeDef::ClearInputSource*  
TIM clear Input sources This parameter can be a value of [TIMEEx\\_ClearInput\\_Source](#)
- *uint32\_t TIM\_ClearInputConfigTypeDef::ClearInputPolarity*  
TIM Clear Input polarity This parameter can be a value of [TIM\\_ClearInput\\_Polarity](#)
- *uint32\_t TIM\_ClearInputConfigTypeDef::ClearInputPrescaler*  
TIM Clear Input prescaler This parameter can be a value of [TIM\\_ClearInput\\_Prescaler](#)
- *uint32\_t TIM\_ClearInputConfigTypeDef::ClearInputFilter*  
TIM Clear Input filter This parameter can be a number between Min\_Data = 0x0 and Max\_Data = 0xFU

### 51.1.8 TIM\_SlaveConfigTypeDef

#### Data Fields

- *uint32\_t SlaveMode*
- *uint32\_t InputTrigger*
- *uint32\_t TriggerPolarity*
- *uint32\_t TriggerPrescaler*
- *uint32\_t TriggerFilter*

#### Field Documentation

- *uint32\_t TIM\_SlaveConfigTypeDef::SlaveMode*  
Slave mode selection This parameter can be a value of [TIMEEx\\_Slave\\_Mode](#)
- *uint32\_t TIM\_SlaveConfigTypeDef::InputTrigger*  
Input Trigger source This parameter can be a value of [TIM\\_Trigger\\_Selection](#)
- *uint32\_t TIM\_SlaveConfigTypeDef::TriggerPolarity*  
Input Trigger polarity This parameter can be a value of [TIM\\_Trigger\\_Polarity](#)
- *uint32\_t TIM\_SlaveConfigTypeDef::TriggerPrescaler*  
Input trigger prescaler This parameter can be a value of [TIM\\_Trigger\\_Prescaler](#)
- *uint32\_t TIM\_SlaveConfigTypeDef::TriggerFilter*  
Input trigger filter This parameter can be a number between Min\_Data = 0x0 and Max\_Data = 0xFU

### 51.1.9 TIM\_HandleTypeDef

#### Data Fields

- *TIM\_TypeDef \* Instance*
- *TIM\_Base\_InitTypeDef Init*
- *HAL\_TIM\_ActiveChannel Channel*
- *DMA\_HandleTypeDef \* hdma*
- *HAL\_LockTypeDef Lock*
- *\_\_IO HAL\_TIM\_StateTypeDef State*

#### Field Documentation

- *TIM\_TypeDef\* TIM\_HandleTypeDef::Instance*  
Register base address
- *TIM\_Base\_InitTypeDef TIM\_HandleTypeDef::Init*  
TIM Time Base required parameters

- ***HAL\_TIM\_ActiveChannel TIM\_HandleTypeDef::Channel***  
Active channel
- ***DMA\_HandleTypeDef\* TIM\_HandleTypeDef::hdma[7]***  
DMA Handlers array This array is accessed by a ***TIM\_DMA\_Handle\_index***
- ***HAL\_LockTypeDef TIM\_HandleTypeDef::Lock***  
Locking object
- ***\_\_IO HAL\_TIM\_StateTypeDef TIM\_HandleTypeDef::State***  
TIM operation state

## 51.2 TIM Firmware driver API description

### 51.2.1 TIMER Generic features

The Timer features include:

1. 16-bit up, down, up/down auto-reload counter.
2. 16-bit programmable prescaler allowing dividing (also on the fly) the counter clock frequency either by any factor between 1 and 65536.
3. Up to 4 independent channels for:
  - Input Capture
  - Output Compare
  - PWM generation (Edge and Center-aligned Mode)
  - One-pulse mode output

### 51.2.2 How to use this driver

1. Initialize the TIM low level resources by implementing the following functions depending from feature used :
  - Time Base : `HAL_TIM_Base_MspInit()`
  - Input Capture : `HAL_TIM_IC_MspInit()`
  - Output Compare : `HAL_TIM_OC_MspInit()`
  - PWM generation : `HAL_TIM_PWM_MspInit()`
  - One-pulse mode output : `HAL_TIM_OnePulse_MspInit()`
  - Encoder mode output : `HAL_TIM_Encoder_MspInit()`
2. Initialize the TIM low level resources :
  - a. Enable the TIM interface clock using `__HAL_RCC_TIMx_CLK_ENABLE()`;
  - b. TIM pins configuration
    - Enable the clock for the TIM GPIOs using the following function:  
`__HAL_RCC_GPIOx_CLK_ENABLE();`
    - Configure these TIM pins in Alternate function mode using `HAL_GPIO_Init()`;
3. The external Clock can be configured, if needed (the default clock is the internal clock from the APBx), using the following function: `HAL_TIM_ConfigClockSource`, the clock configuration should be done before any start function.
4. Configure the TIM in the desired functioning mode using one of the Initialization function of this driver:
  - `HAL_TIM_Base_Init`: to use the Timer to generate a simple time base
  - `HAL_TIM_OC_Init` and `HAL_TIM_OC_ConfigChannel`: to use the Timer to generate an Output Compare signal.
  - `HAL_TIM_PWM_Init` and `HAL_TIM_PWM_ConfigChannel`: to use the Timer to generate a PWM signal.
  - `HAL_TIM_IC_Init` and `HAL_TIM_IC_ConfigChannel`: to use the Timer to measure an external signal.
  - `HAL_TIM_OnePulse_Init` and `HAL_TIM_OnePulse_ConfigChannel`: to use the Timer in One Pulse Mode.
  - `HAL_TIM_Encoder_Init`: to use the Timer Encoder Interface.

5. Activate the TIM peripheral using one of the start functions depending from the feature used:
  - Time Base : HAL\_TIM\_Base\_Start(), HAL\_TIM\_Base\_Start\_DMA(),  
HAL\_TIM\_Base\_Start\_IT()
  - Input Capture : HAL\_TIM\_IC\_Start(), HAL\_TIM\_IC\_Start\_DMA(),  
HAL\_TIM\_IC\_Start\_IT()
  - Output Compare : HAL\_TIM\_OC\_Start(), HAL\_TIM\_OC\_Start\_DMA(),  
HAL\_TIM\_OC\_Start\_IT()
  - PWM generation : HAL\_TIM\_PWM\_Start(), HAL\_TIM\_PWM\_Start\_DMA(),  
HAL\_TIM\_PWM\_Start\_IT()
  - One-pulse mode output : HAL\_TIM\_OnePulse\_Start(),  
HAL\_TIM\_OnePulse\_Start\_IT()
  - Encoder mode output : HAL\_TIM\_Encoder\_Start(),  
HAL\_TIM\_Encoder\_Start\_DMA(), HAL\_TIM\_Encoder\_Start\_IT().
6. The DMA Burst is managed with the two following functions:  
HAL\_TIM\_DMABurst\_WriteStart() HAL\_TIM\_DMABurst\_ReadStart()

### 51.2.3 Time Base functions

This section provides functions allowing to:

- Initialize and configure the TIM base.
- De-initialize the TIM base.
- Start the Time Base.
- Stop the Time Base.
- Start the Time Base and enable interrupt.
- Stop the Time Base and disable interrupt.
- Start the Time Base and enable DMA transfer.
- Stop the Time Base and disable DMA transfer.

This section contains the following APIs:

- [\*HAL\\_TIM\\_Base\\_Init\(\)\*](#)
- [\*HAL\\_TIM\\_Base\\_DeInit\(\)\*](#)
- [\*HAL\\_TIM\\_Base\\_MspInit\(\)\*](#)
- [\*HAL\\_TIM\\_Base\\_MspDeInit\(\)\*](#)
- [\*HAL\\_TIM\\_Base\\_Start\(\)\*](#)
- [\*HAL\\_TIM\\_Base\\_Stop\(\)\*](#)
- [\*HAL\\_TIM\\_Base\\_Start\\_IT\(\)\*](#)
- [\*HAL\\_TIM\\_Base\\_Stop\\_IT\(\)\*](#)
- [\*HAL\\_TIM\\_Base\\_Start\\_DMA\(\)\*](#)
- [\*HAL\\_TIM\\_Base\\_Stop\\_DMA\(\)\*](#)

### 51.2.4 Time Output Compare functions

This section provides functions allowing to:

- Initialize and configure the TIM Output Compare.
- De-initialize the TIM Output Compare.
- Start the Time Output Compare.
- Stop the Time Output Compare.
- Start the Time Output Compare and enable interrupt.
- Stop the Time Output Compare and disable interrupt.
- Start the Time Output Compare and enable DMA transfer.
- Stop the Time Output Compare and disable DMA transfer.

This section contains the following APIs:

- [`HAL\_TIM\_OC\_Init\(\)`](#)
- [`HAL\_TIM\_OC\_DelInit\(\)`](#)
- [`HAL\_TIM\_OC\_MspInit\(\)`](#)
- [`HAL\_TIM\_OC\_MspDelInit\(\)`](#)
- [`HAL\_TIM\_OC\_Start\(\)`](#)
- [`HAL\_TIM\_OC\_Stop\(\)`](#)
- [`HAL\_TIM\_OC\_Start\_IT\(\)`](#)
- [`HAL\_TIM\_OC\_Stop\_IT\(\)`](#)
- [`HAL\_TIM\_OC\_Start\_DMA\(\)`](#)
- [`HAL\_TIM\_OC\_Stop\_DMA\(\)`](#)

### 51.2.5 Time PWM functions

This section provides functions allowing to:

- Initialize and configure the TIM OPWM.
- De-initialize the TIM PWM.
- Start the Time PWM.
- Stop the Time PWM.
- Start the Time PWM and enable interrupt.
- Stop the Time PWM and disable interrupt.
- Start the Time PWM and enable DMA transfer.
- Stop the Time PWM and disable DMA transfer.

This section contains the following APIs:

- [`HAL\_TIM\_PWM\_Init\(\)`](#)
- [`HAL\_TIM\_PWM\_DelInit\(\)`](#)
- [`HAL\_TIM\_PWM\_MspInit\(\)`](#)
- [`HAL\_TIM\_PWM\_MspDelInit\(\)`](#)
- [`HAL\_TIM\_PWM\_Start\(\)`](#)
- [`HAL\_TIM\_PWM\_Stop\(\)`](#)
- [`HAL\_TIM\_PWM\_Start\_IT\(\)`](#)
- [`HAL\_TIM\_PWM\_Stop\_IT\(\)`](#)
- [`HAL\_TIM\_PWM\_Start\_DMA\(\)`](#)
- [`HAL\_TIM\_PWM\_Stop\_DMA\(\)`](#)

### 51.2.6 Time Input Capture functions

This section provides functions allowing to:

- Initialize and configure the TIM Input Capture.
- De-initialize the TIM Input Capture.
- Start the Time Input Capture.
- Stop the Time Input Capture.
- Start the Time Input Capture and enable interrupt.
- Stop the Time Input Capture and disable interrupt.
- Start the Time Input Capture and enable DMA transfer.
- Stop the Time Input Capture and disable DMA transfer.

This section contains the following APIs:

- [`HAL\_TIM\_IC\_Init\(\)`](#)
- [`HAL\_TIM\_IC\_DelInit\(\)`](#)
- [`HAL\_TIM\_IC\_MspInit\(\)`](#)

- [`HAL\_TIM\_IC\_MspDeInit\(\)`](#)
- [`HAL\_TIM\_IC\_Start\(\)`](#)
- [`HAL\_TIM\_IC\_Stop\(\)`](#)
- [`HAL\_TIM\_IC\_Start\_IT\(\)`](#)
- [`HAL\_TIM\_IC\_Stop\_IT\(\)`](#)
- [`HAL\_TIM\_IC\_Start\_DMA\(\)`](#)
- [`HAL\_TIM\_IC\_Stop\_DMA\(\)`](#)

### 51.2.7 Time One Pulse functions

This section provides functions allowing to:

- Initialize and configure the TIM One Pulse.
- De-initialize the TIM One Pulse.
- Start the Time One Pulse.
- Stop the Time One Pulse.
- Start the Time One Pulse and enable interrupt.
- Stop the Time One Pulse and disable interrupt.
- Start the Time One Pulse and enable DMA transfer.
- Stop the Time One Pulse and disable DMA transfer.

This section contains the following APIs:

- [`HAL\_TIM\_OnePulse\_Init\(\)`](#)
- [`HAL\_TIM\_OnePulse\_DeInit\(\)`](#)
- [`HAL\_TIM\_OnePulse\_MspInit\(\)`](#)
- [`HAL\_TIM\_OnePulse\_MspDeInit\(\)`](#)
- [`HAL\_TIM\_OnePulse\_Start\(\)`](#)
- [`HAL\_TIM\_OnePulse\_Stop\(\)`](#)
- [`HAL\_TIM\_OnePulse\_Start\_IT\(\)`](#)
- [`HAL\_TIM\_OnePulse\_Stop\_IT\(\)`](#)

### 51.2.8 Time Encoder functions

This section provides functions allowing to:

- Initialize and configure the TIM Encoder.
- De-initialize the TIM Encoder.
- Start the Time Encoder.
- Stop the Time Encoder.
- Start the Time Encoder and enable interrupt.
- Stop the Time Encoder and disable interrupt.
- Start the Time Encoder and enable DMA transfer.
- Stop the Time Encoder and disable DMA transfer.

This section contains the following APIs:

- [`HAL\_TIM\_Encoder\_Init\(\)`](#)
- [`HAL\_TIM\_Encoder\_DeInit\(\)`](#)
- [`HAL\_TIM\_Encoder\_MspInit\(\)`](#)
- [`HAL\_TIM\_Encoder\_MspDeInit\(\)`](#)
- [`HAL\_TIM\_Encoder\_Start\(\)`](#)
- [`HAL\_TIM\_Encoder\_Stop\(\)`](#)
- [`HAL\_TIM\_Encoder\_Start\_IT\(\)`](#)
- [`HAL\_TIM\_Encoder\_Stop\_IT\(\)`](#)
- [`HAL\_TIM\_Encoder\_Start\_DMA\(\)`](#)
- [`HAL\_TIM\_Encoder\_Stop\_DMA\(\)`](#)

### 51.2.9 IRQ handler management

This section provides Timer IRQ handler function.

This section contains the following APIs:

- [\*HAL\\_TIM\\_IRQHandler\(\)\*](#)

### 51.2.10 Peripheral Control functions

This section provides functions allowing to:

- Configure The Input Output channels for OC, PWM, IC or One Pulse mode.
- Configure External Clock source.
- Configure Complementary channels, break features and dead time.
- Configure Master and the Slave synchronization.
- Configure the DMA Burst Mode.

This section contains the following APIs:

- [\*HAL\\_TIM\\_OC\\_ConfigChannel\(\)\*](#)
- [\*HAL\\_TIM\\_IC\\_ConfigChannel\(\)\*](#)
- [\*HAL\\_TIM\\_PWM\\_ConfigChannel\(\)\*](#)
- [\*HAL\\_TIM\\_OnePulse\\_ConfigChannel\(\)\*](#)
- [\*HAL\\_TIM\\_DMABurst\\_WriteStart\(\)\*](#)
- [\*HAL\\_TIM\\_DMABurst\\_MultiWriteStart\(\)\*](#)
- [\*HAL\\_TIM\\_DMABurst\\_WriteStop\(\)\*](#)
- [\*HAL\\_TIM\\_DMABurst\\_ReadStart\(\)\*](#)
- [\*HAL\\_TIM\\_DMABurst\\_MultiReadStart\(\)\*](#)
- [\*HAL\\_TIM\\_DMABurst\\_ReadStop\(\)\*](#)
- [\*HAL\\_TIM\\_GenerateEvent\(\)\*](#)
- [\*HAL\\_TIM\\_ConfigOCrefClear\(\)\*](#)
- [\*HAL\\_TIM\\_ConfigClockSource\(\)\*](#)
- [\*HAL\\_TIM\\_ConfigTI1Input\(\)\*](#)
- [\*HAL\\_TIM\\_SlaveConfigSynchronization\(\)\*](#)
- [\*HAL\\_TIM\\_SlaveConfigSynchronization\\_IT\(\)\*](#)
- [\*HAL\\_TIM\\_ReadCapturedValue\(\)\*](#)

### 51.2.11 TIM Callbacks functions

This section provides TIM callback functions:

- Timer Period elapsed callback
- Timer Output Compare callback
- Timer Input capture callback
- Timer Trigger callback
- Timer Error callback

This section contains the following APIs:

- [\*HAL\\_TIM\\_PeriodElapsedCallback\(\)\*](#)
- [\*HAL\\_TIM\\_OC\\_DelayElapsedCallback\(\)\*](#)
- [\*HAL\\_TIM\\_IC\\_CaptureCallback\(\)\*](#)
- [\*HAL\\_TIM\\_PWM\\_PulseFinishedCallback\(\)\*](#)
- [\*HAL\\_TIM\\_TriggerCallback\(\)\*](#)
- [\*HAL\\_TIM\\_ErrorCallback\(\)\*](#)

### 51.2.12 Peripheral State functions

This subsection permit to get in run-time the status of the peripheral and the data flow.

This section contains the following APIs:

- [`HAL\_TIM\_Base\_GetState\(\)`](#)
- [`HAL\_TIM\_OC\_GetState\(\)`](#)
- [`HAL\_TIM\_PWM\_GetState\(\)`](#)
- [`HAL\_TIM\_IC\_GetState\(\)`](#)
- [`HAL\_TIM\_OnePulse\_GetState\(\)`](#)
- [`HAL\_TIM\_Encoder\_GetState\(\)`](#)

### 51.2.13 Detailed description of functions

#### `HAL_TIM_Base_Init`

|                      |                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_TIM_Base_Init (TIM_HandleTypeDef * htim)</code>                                                                      |
| Function description | Initializes the TIM Time base Unit according to the specified parameters in the <code>TIM_HandleTypeDef</code> and create the associated handle. |
| Parameters           | <ul style="list-style-type: none"><li>• <code>htim</code>: TIM Base handle</li></ul>                                                             |
| Return values        | <ul style="list-style-type: none"><li>• <code>HAL</code>: status</li></ul>                                                                       |

#### `HAL_TIM_Base_DeInit`

|                      |                                                                                      |
|----------------------|--------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_TIM_Base_DeInit (TIM_HandleTypeDef * htim)</code>        |
| Function description | DeInitializes the TIM Base peripheral.                                               |
| Parameters           | <ul style="list-style-type: none"><li>• <code>htim</code>: TIM Base handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <code>HAL</code>: status</li></ul>           |

#### `HAL_TIM_Base_MspInit`

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <code>void HAL_TIM_Base_MspInit (TIM_HandleTypeDef * htim)</code>               |
| Function description | Initializes the TIM Base MSP.                                                   |
| Parameters           | <ul style="list-style-type: none"><li>• <code>htim</code>: TIM handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <code>None</code></li></ul>             |

#### `HAL_TIM_Base_MspDeInit`

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <code>void HAL_TIM_Base_MspDeInit (TIM_HandleTypeDef * htim)</code>             |
| Function description | DeInitializes TIM Base MSP.                                                     |
| Parameters           | <ul style="list-style-type: none"><li>• <code>htim</code>: TIM handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <code>None</code></li></ul>             |

**HAL\_TIM\_Base\_Start**

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_Base_Start<br/>(TIM_HandleTypeDef * htim)</b> |
| Function description | Starts the TIM Base generation.                                            |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM handle</li></ul>  |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>       |

**HAL\_TIM\_Base\_Stop**

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_Base_Stop<br/>(TIM_HandleTypeDef * htim)</b> |
| Function description | Stops the TIM Base generation.                                            |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>      |

**HAL\_TIM\_Base\_Start\_IT**

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_Base_Start_IT<br/>(TIM_HandleTypeDef * htim)</b> |
| Function description | Starts the TIM Base generation in interrupt mode.                             |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM handle</li></ul>     |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>          |

**HAL\_TIM\_Base\_Stop\_IT**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_Base_Stop_IT<br/>(TIM_HandleTypeDef * htim)</b> |
| Function description | Stops the TIM Base generation in interrupt mode.                             |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM handle</li></ul>    |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>         |

**HAL\_TIM\_Base\_Start\_DMA**

|                      |                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_Base_Start_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t * pData, uint16_t Length)</b>                                                                                                     |
| Function description | Starts the TIM Base generation in DMA mode.                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM handle</li><li>• <b>pData:</b> The source Buffer address.</li><li>• <b>Length:</b> The length of data to be transferred from memory to peripheral.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                                                                  |

**HAL\_TIM\_Base\_Stop\_DMA**

Function name      **HAL\_StatusTypeDef HAL\_TIM\_Base\_Stop\_DMA  
(TIM\_HandleTypeDef \* htim)**

Function description      Stops the TIM Base generation in DMA mode.

Parameters      • **htim:** TIM handle

Return values      • **HAL:** status

**HAL\_TIM\_OC\_Init**

Function name      **HAL\_StatusTypeDef HAL\_TIM\_OC\_Init (TIM\_HandleTypeDef \* htim)**

Function description      Initializes the TIM Output Compare according to the specified parameters in the TIM\_HandleTypeDef and create the associated handle.

Parameters      • **htim:** TIM Output Compare handle

Return values      • **HAL:** status

**HAL\_TIM\_OC\_DelInit**

Function name      **HAL\_StatusTypeDef HAL\_TIM\_OC\_DelInit  
(TIM\_HandleTypeDef \* htim)**

Function description      Delinitializes the TIM peripheral.

Parameters      • **htim:** TIM Output Compare handle

Return values      • **HAL:** status

**HAL\_TIM\_OC\_MspInit**

Function name      **void HAL\_TIM\_OC\_MspInit (TIM\_HandleTypeDef \* htim)**

Function description      Initializes the TIM Output Compare MSP.

Parameters      • **htim:** TIM handle

Return values      • **None**

**HAL\_TIM\_OC\_MspDelInit**

Function name      **void HAL\_TIM\_OC\_MspDelInit (TIM\_HandleTypeDef \* htim)**

Function description      Delinitializes TIM Output Compare MSP.

Parameters      • **htim:** TIM handle

Return values      • **None**

**HAL\_TIM\_OC\_Start**

Function name      **HAL\_StatusTypeDef HAL\_TIM\_OC\_Start (TIM\_HandleTypeDef  
\* htim, uint32\_t Channel)**

Function description      Starts the TIM Output Compare signal generation.

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li><b>htim:</b> TIM Output Compare handle</li> <li><b>Channel:</b> TIM Channel to be enabled This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                           |

### HAL\_TIM\_OC\_Stop

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_OC_Stop (TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                            |
| Function description | Stops the TIM Output Compare signal generation.                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>htim:</b> TIM handle</li> <li><b>Channel:</b> TIM Channel to be disabled This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                             |

### HAL\_TIM\_OC\_Start\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_OC_Start_IT (TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                          |
| Function description | Starts the TIM Output Compare signal generation in interrupt mode.                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li><b>htim:</b> TIM OC handle</li> <li><b>Channel:</b> TIM Channel to be enabled This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                               |

### HAL\_TIM\_OC\_Stop\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_OC_Stop_IT (TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                          |
| Function description | Stops the TIM Output Compare signal generation in interrupt mode.                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li><b>htim:</b> TIM Output Compare handle</li> <li><b>Channel:</b> TIM Channel to be disabled This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>TIM_CHANNEL_2: TIM Channel 2 selected</li> </ul> </li> </ul> |

|               |                                                                                                                                            |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>- TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>- TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                     |

### HAL\_TIM\_OC\_Start\_DMA

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_OC_Start_DMA</b><br><b>(TIM_HandleTypeDef * htim, uint32_t Channel, uint32_t * pData, uint16_t Length)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Starts the TIM Output Compare signal generation in DMA mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Output Compare handle</li> <li>• <b>Channel:</b> TIM Channel to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>- TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>- TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>- TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> <li>• <b>pData:</b> The source Buffer address.</li> <li>• <b>Length:</b> The length of data to be transferred from memory to TIM peripheral</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

### HAL\_TIM\_OC\_Stop\_DMA

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_OC_Stop_DMA</b><br><b>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                               |
| Function description | Stops the TIM Output Compare signal generation in DMA mode.                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Output Compare handle</li> <li>• <b>Channel:</b> TIM Channel to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>- TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>- TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>- TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                            |

### HAL\_TIM\_PWM\_Init

|                      |                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_PWM_Init</b><br><b>(TIM_HandleTypeDef * htim)</b>                                                     |
| Function description | Initializes the TIM PWM Time Base according to the specified parameters in the TIM_HandleTypeDef and create the associated handle. |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> </ul>                                                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                             |

**HAL\_TIM\_PWM\_DeInit**

Function name      **HAL\_StatusTypeDef HAL\_TIM\_PWM\_DeInit  
(TIM\_HandleTypeDef \* htim)**

Function description      DeInitializes the TIM peripheral.

Parameters      • **htim:** TIM handle

Return values      • **HAL:** status

**HAL\_TIM\_PWM\_MspInit**

Function name      **void HAL\_TIM\_PWM\_MspInit (TIM\_HandleTypeDef \* htim)**

Function description      Initializes the TIM PWM MSP.

Parameters      • **htim:** TIM handle

Return values      • **None**

**HAL\_TIM\_PWM\_MspDeInit**

Function name      **void HAL\_TIM\_PWM\_MspDeInit (TIM\_HandleTypeDef \* htim)**

Function description      DeInitializes TIM PWM MSP.

Parameters      • **htim:** TIM handle

Return values      • **None**

**HAL\_TIM\_PWM\_Start**

Function name      **HAL\_StatusTypeDef HAL\_TIM\_PWM\_Start  
(TIM\_HandleTypeDef \* htim, uint32\_t Channel)**

Function description      Starts the PWM signal generation.

Parameters      • **htim:** TIM handle  
• **Channel:** TIM Channels to be enabled This parameter can be one of the following values:  
– TIM\_CHANNEL\_1: TIM Channel 1 selected  
– TIM\_CHANNEL\_2: TIM Channel 2 selected  
– TIM\_CHANNEL\_3: TIM Channel 3 selected  
– TIM\_CHANNEL\_4: TIM Channel 4 selected

Return values      • **HAL:** status

**HAL\_TIM\_PWM\_Stop**

Function name      **HAL\_StatusTypeDef HAL\_TIM\_PWM\_Stop  
(TIM\_HandleTypeDef \* htim, uint32\_t Channel)**

Function description      Stops the PWM signal generation.

Parameters      • **htim:** TIM handle  
• **Channel:** TIM Channels to be disabled This parameter can be one of the following values:  
– TIM\_CHANNEL\_1: TIM Channel 1 selected  
– TIM\_CHANNEL\_2: TIM Channel 2 selected  
– TIM\_CHANNEL\_3: TIM Channel 3 selected

|               |                                                                                                  |
|---------------|--------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>- <b>TIM_CHANNEL_4:</b> TIM Channel 4 selected</li> </ul> |
|               | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                           |

### HAL\_TIM\_PWM\_Start\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_PWM_Start_IT<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Starts the PWM signal generation in interrupt mode.                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <b>TIM_CHANNEL_1:</b> TIM Channel 1 selected</li> <li>- <b>TIM_CHANNEL_2:</b> TIM Channel 2 selected</li> <li>- <b>TIM_CHANNEL_3:</b> TIM Channel 3 selected</li> <li>- <b>TIM_CHANNEL_4:</b> TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                         |

### HAL\_TIM\_PWM\_Stop\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_PWM_Stop_IT<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Stops the PWM signal generation in interrupt mode.                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channels to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <b>TIM_CHANNEL_1:</b> TIM Channel 1 selected</li> <li>- <b>TIM_CHANNEL_2:</b> TIM Channel 2 selected</li> <li>- <b>TIM_CHANNEL_3:</b> TIM Channel 3 selected</li> <li>- <b>TIM_CHANNEL_4:</b> TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                          |

### HAL\_TIM\_PWM\_Start\_DMA

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_PWM_Start_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t Channel, uint32_t * pData, uint16_t Length)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Starts the TIM PWM signal generation in DMA mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <b>TIM_CHANNEL_1:</b> TIM Channel 1 selected</li> <li>- <b>TIM_CHANNEL_2:</b> TIM Channel 2 selected</li> <li>- <b>TIM_CHANNEL_3:</b> TIM Channel 3 selected</li> <li>- <b>TIM_CHANNEL_4:</b> TIM Channel 4 selected</li> </ul> </li> <li>• <b>pData:</b> The source Buffer address.</li> <li>• <b>Length:</b> The length of data to be transferred from memory to TIM peripheral</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

**HAL\_TIM\_PWM\_Stop\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_PWM_Stop_DMA (TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                          |
| Function description | Stops the TIM PWM signal generation in DMA mode.                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channels to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>- TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>- TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>- TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                              |

**HAL\_TIM\_IC\_Init**

|                      |                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_IC_Init (TIM_HandleTypeDef * htim)</b>                                                                          |
| Function description | Initializes the TIM Input Capture Time base according to the specified parameters in the TIM_HandleTypeDef and create the associated handle. |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Input Capture handle</li> </ul>                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                       |

**HAL\_TIM\_IC\_DelInit**

|                      |                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_IC_DelInit (TIM_HandleTypeDef * htim)</b>                    |
| Function description | Deinitializes the TIM peripheral.                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Input Capture handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                    |

**HAL\_TIM\_IC\_MspInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_TIM_IC_MspInit (TIM_HandleTypeDef * htim)</b>                   |
| Function description | Initializes the TIM Input Capture MSP.                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_TIM\_IC\_MspDelInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_TIM_IC_MspDelInit (TIM_HandleTypeDef * htim)</b>                |
| Function description | Deinitializes TIM Input Capture MSP.                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_TIM\_IC\_Start**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_IC_Start (TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Starts the TIM Input Capture measurement.                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Input Capture handle</li> <li>• <b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                           |

**HAL\_TIM\_IC\_Stop**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_IC_Stop (TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                               |
| Function description | Stops the TIM Input Capture measurement.                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channels to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                              |

**HAL\_TIM\_IC\_Start\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_IC_Start_IT (TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Starts the TIM Input Capture measurement in interrupt mode.                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Input Capture handle</li> <li>• <b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                           |

**HAL\_TIM\_IC\_Stop\_IT**

|                      |                                                                                          |
|----------------------|------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_IC_Stop_IT (TIM_HandleTypeDef * htim, uint32_t Channel)</b> |
| Function description | Stops the TIM Input Capture measurement in interrupt mode.                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> </ul>              |

- **Channel:** TIM Channels to be disabled This parameter can be one of the following values:
  - TIM\_CHANNEL\_1: TIM Channel 1 selected
  - TIM\_CHANNEL\_2: TIM Channel 2 selected
  - TIM\_CHANNEL\_3: TIM Channel 3 selected
  - TIM\_CHANNEL\_4: TIM Channel 4 selected

Return values

- **HAL:** status

### HAL\_TIM\_IC\_Start\_DMA

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_IC_Start_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t Channel, uint32_t * pData, uint16_t Length)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Starts the TIM Input Capture measurement in DMA mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Input Capture handle</li> <li>• <b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> <li>• <b>pData:</b> The destination Buffer address.</li> <li>• <b>Length:</b> The length of data to be transferred from TIM peripheral to memory.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

### HAL\_TIM\_IC\_Stop\_DMA

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_IC_Stop_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Stops the TIM Input Capture measurement in DMA mode.                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Input Capture handle</li> <li>• <b>Channel:</b> TIM Channels to be disabled This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                      |

### HAL\_TIM\_OnePulse\_Init

|                      |                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_OnePulse_Init<br/>(TIM_HandleTypeDef * htim, uint32_t OnePulseMode)</b>                                                           |
| Function description | Initializes the TIM One Pulse Time Base according to the specified parameters in the TIM_HandleTypeDef and create the associated handle.                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM OnePulse handle</li> <li>• <b>OnePulseMode:</b> Select the One pulse mode. This parameter</li> </ul> |

can be one of the following values:

- TIM\_OPMODE\_SINGLE: Only one pulse will be generated.
- TIM\_OPMODE\_REPETITIVE: Repetitive pulses will be generated.

Return values

- **HAL:** status

### **HAL\_TIM\_OnePulse\_DelInit**

Function name

**HAL\_StatusTypeDef HAL\_TIM\_OnePulse\_DelInit  
(TIM\_HandleTypeDef \* htim)**

Function description

DeInitializes the TIM One Pulse.

Parameters

- **htim:** TIM One Pulse handle

Return values

- **HAL:** status

### **HAL\_TIM\_OnePulse\_MspInit**

Function name

**void HAL\_TIM\_OnePulse\_MspInit (TIM\_HandleTypeDef \* htim)**

Function description

Initializes the TIM One Pulse MSP.

Parameters

- **htim:** TIM handle

Return values

- **None**

### **HAL\_TIM\_OnePulse\_MspDelInit**

Function name

**void HAL\_TIM\_OnePulse\_MspDelInit (TIM\_HandleTypeDef \* htim)**

Function description

DeInitializes TIM One Pulse MSP.

Parameters

- **htim:** TIM handle

Return values

- **None**

### **HAL\_TIM\_OnePulse\_Start**

Function name

**HAL\_StatusTypeDef HAL\_TIM\_OnePulse\_Start  
(TIM\_HandleTypeDef \* htim, uint32\_t OutputChannel)**

Function description

Starts the TIM One Pulse signal generation.

Parameters

- **htim:** TIM One Pulse handle
- **OutputChannel:** TIM Channels to be enabled This parameter can be one of the following values:
  - TIM\_CHANNEL\_1: TIM Channel 1 selected
  - TIM\_CHANNEL\_2: TIM Channel 2 selected

Return values

- **HAL:** status

### **HAL\_TIM\_OnePulse\_Stop**

Function name

**HAL\_StatusTypeDef HAL\_TIM\_OnePulse\_Stop  
(TIM\_HandleTypeDef \* htim, uint32\_t OutputChannel)**

|                      |                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function description | Stops the TIM One Pulse signal generation.                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM One Pulse handle</li> <li>• <b>OutputChannel:</b> TIM Channels to be disable This parameter can be one of the following values:               <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                         |

### **HAL\_TIM\_OnePulse\_Start\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_OnePulse_Start_IT<br/>(TIM_HandleTypeDef * htim, uint32_t OutputChannel)</b>                                                                                                                                                                                                                                                      |
| Function description | Starts the TIM One Pulse signal generation in interrupt mode.                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM One Pulse handle</li> <li>• <b>OutputChannel:</b> TIM Channels to be enabled This parameter can be one of the following values:               <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                         |

### **HAL\_TIM\_OnePulse\_Stop\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_OnePulse_Stop_IT<br/>(TIM_HandleTypeDef * htim, uint32_t OutputChannel)</b>                                                                                                                                                                                                                                                       |
| Function description | Stops the TIM One Pulse signal generation in interrupt mode.                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM One Pulse handle</li> <li>• <b>OutputChannel:</b> TIM Channels to be enabled This parameter can be one of the following values:               <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                         |

### **HAL\_TIM\_Encoder\_Init**

|                      |                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_Encoder_Init<br/>(TIM_HandleTypeDef * htim, TIM_Encoder_InitTypeDef * sConfig)</b>                                                        |
| Function description | Initializes the TIM Encoder Interface and create the associated handle.                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Encoder Interface handle</li> <li>• <b>sConfig:</b> TIM Encoder Interface configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                 |

**HAL\_TIM\_Encoder\_DelInit**

Function name      **HAL\_StatusTypeDef HAL\_TIM\_Encoder\_DelInit  
(TIM\_HandleTypeDef \* htim)**

Function description      DelInitializes the TIM Encoder interface.

Parameters      • **htim:** TIM Encoder handle

Return values      • **HAL:** status

**HAL\_TIM\_Encoder\_MspInit**

Function name      **void HAL\_TIM\_Encoder\_MspInit (TIM\_HandleTypeDef \* htim)**

Function description      Initializes the TIM Encoder Interface MSP.

Parameters      • **htim:** TIM handle

Return values      • **None**

**HAL\_TIM\_Encoder\_MspDeInit**

Function name      **void HAL\_TIM\_Encoder\_MspDeInit (TIM\_HandleTypeDef \* htim)**

Function description      DelInitializes TIM Encoder Interface MSP.

Parameters      • **htim:** TIM handle

Return values      • **None**

**HAL\_TIM\_Encoder\_Start**

Function name      **HAL\_StatusTypeDef HAL\_TIM\_Encoder\_Start  
(TIM\_HandleTypeDef \* htim, uint32\_t Channel)**

Function description      Starts the TIM Encoder Interface.

Parameters      • **htim:** TIM Encoder Interface handle  
• **Channel:** TIM Channels to be enabled This parameter can be one of the following values:  
– **TIM\_CHANNEL\_1:** TIM Channel 1 selected  
– **TIM\_CHANNEL\_2:** TIM Channel 2 selected  
– **TIM\_CHANNEL\_ALL:** TIM Channel 1 and TIM Channel 2 are selected

Return values      • **HAL:** status

**HAL\_TIM\_Encoder\_Stop**

Function name      **HAL\_StatusTypeDef HAL\_TIM\_Encoder\_Stop  
(TIM\_HandleTypeDef \* htim, uint32\_t Channel)**

Function description      Stops the TIM Encoder Interface.

Parameters      • **htim:** TIM Encoder Interface handle  
• **Channel:** TIM Channels to be disabled This parameter can be one of the following values:  
– **TIM\_CHANNEL\_1:** TIM Channel 1 selected  
– **TIM\_CHANNEL\_2:** TIM Channel 2 selected

|               |                                                                                                                   |
|---------------|-------------------------------------------------------------------------------------------------------------------|
|               | <ul style="list-style-type: none"> <li>- TIM_CHANNEL_ALL: TIM Channel 1 and TIM Channel 2 are selected</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                            |

### HAL\_TIM\_Encoder\_Start\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_Encoder_Start_IT<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                          |
| Function description | Starts the TIM Encoder Interface in interrupt mode.                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Encoder Interface handle</li> <li>• <b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>- TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>- TIM_CHANNEL_ALL: TIM Channel 1 and TIM Channel 2 are selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                      |

### HAL\_TIM\_Encoder\_Stop\_IT

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_Encoder_Stop_IT<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                            |
| Function description | Stops the TIM Encoder Interface in interrupt mode.                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Encoder Interface handle</li> <li>• <b>Channel:</b> TIM Channels to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>- TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>- TIM_CHANNEL_ALL: TIM Channel 1 and TIM Channel 2 are selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                       |

### HAL\_TIM\_Encoder\_Start\_DMA

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_Encoder_Start_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t Channel, uint32_t *<br/>pData1, uint32_t * pData2, uint16_t Length)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Starts the TIM Encoder Interface in DMA mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Encoder Interface handle</li> <li>• <b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>- TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>- TIM_CHANNEL_ALL: TIM Channel 1 and TIM Channel 2 are selected</li> </ul> </li> <li>• <b>pData1:</b> The destination Buffer address for IC1.</li> <li>• <b>pData2:</b> The destination Buffer address for IC2.</li> <li>• <b>Length:</b> The length of data to be transferred from TIM</li> </ul> |

peripheral to memory.

|               |                                                                      |
|---------------|----------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul> |
|---------------|----------------------------------------------------------------------|

### HAL\_TIM\_Encoder\_Stop\_DMA

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_Encoder_Stop_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                |
| Function description | Stops the TIM Encoder Interface in DMA mode.                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>htim:</b> TIM Encoder Interface handle</li> <li><b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>TIM_CHANNEL_ALL: TIM Channel 1 and TIM Channel 2 are selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                              |

### HAL\_TIM\_IRQHandler

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>void HAL_TIM_IRQHandler (TIM_HandleTypeDef * htim)</b>                 |
| Function description | This function handles TIM interrupts requests.                            |
| Parameters           | <ul style="list-style-type: none"> <li><b>htim:</b> TIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>             |

### HAL\_TIM\_OC\_ConfigChannel

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_OC_ConfigChannel<br/>(TIM_HandleTypeDef * htim, TIM_OC_InitTypeDef * sConfig,<br/>uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                                      |
| Function description | Initializes the TIM Output Compare Channels according to the specified parameters in the TIM_OC_InitTypeDef.                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>htim:</b> TIM Output Compare handle</li> <li><b>sConfig:</b> TIM Output Compare configuration structure</li> <li><b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                      |

**HAL\_TIM\_PWM\_ConfigChannel**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_PWM_ConfigChannel<br/>(TIM_HandleTypeDef * htim, TIM_OC_InitTypeDef * sConfig,<br/>uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                         |
| Function description | Initializes the TIM PWM channels according to the specified parameters in the TIM_OC_InitTypeDef.                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>sConfig:</b> TIM PWM configuration structure</li> <li>• <b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>- TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>- TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>- TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                        |

**HAL\_TIM\_IC\_ConfigChannel**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_IC_ConfigChannel<br/>(TIM_HandleTypeDef * htim, TIM_IC_InitTypeDef * sConfig,<br/>uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Initializes the TIM Input Capture Channels according to the specified parameters in the TIM_IC_InitTypeDef.                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM IC handle</li> <li>• <b>sConfig:</b> TIM Input Capture configuration structure</li> <li>• <b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>- TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>- TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>- TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                     |

**HAL\_TIM\_OnePulse\_ConfigChannel**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_OnePulse_ConfigChannel<br/>(TIM_HandleTypeDef * htim, TIM_OnePulse_InitTypeDef *<br/>sConfig, uint32_t OutputChannel, uint32_t InputChannel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Initializes the TIM One Pulse Channels according to the specified parameters in the TIM_OnePulse_InitTypeDef.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM One Pulse handle</li> <li>• <b>sConfig:</b> TIM One Pulse configuration structure</li> <li>• <b>OutputChannel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>- TIM_CHANNEL_2: TIM Channel 2 selected</li> </ul> </li> <li>• <b>InputChannel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> </ul> </li> </ul> |

- TIM\_CHANNEL\_2: TIM Channel 2 selected
- **Return values**
- **HAL:** status

### **HAL\_TIM\_ConfigOCrefClear**

|                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>        | <b>HAL_StatusTypeDef HAL_TIM_ConfigOCrefClear<br/>(TIM_HandleTypeDef * htim, TIM_ClearInputConfigTypeDef *<br/>sClearInputConfig, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Function description</b> | Configures the OCRef clear feature.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Parameters</b>           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>sClearInputConfig:</b> pointer to a TIM_ClearInputConfigTypeDef structure that contains the OCREF clear feature and parameters for the TIM peripheral.</li> <li>• <b>Channel:</b> specifies the TIM Channel This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1</li> <li>- TIM_CHANNEL_2: TIM Channel 2</li> <li>- TIM_CHANNEL_3: TIM Channel 3</li> <li>- TIM_CHANNEL_4: TIM Channel 4</li> </ul> </li> <li>• <b>htim:</b> TIM handle</li> <li>• <b>sClearInputConfig:</b> pointer to a TIM_ClearInputConfigTypeDef structure that contains the OCREF clear feature and parameters for the TIM peripheral.</li> <li>• <b>Channel:</b> specifies the TIM Channel This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1</li> <li>- TIM_CHANNEL_2: TIM Channel 2</li> <li>- TIM_CHANNEL_3: TIM Channel 3</li> <li>- TIM_CHANNEL_4: TIM Channel 4</li> <li>- TIM_Channel_5: TIM Channel 5</li> <li>- TIM_Channel_6: TIM Channel 6</li> </ul> </li> </ul> |
| <b>Return values</b>        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Notes</b>                | <ul style="list-style-type: none"> <li>• For STM32F302xC, STM32F302xE, STM32F303xC, STM32F303xE, STM32F358xx, STM32F398xx and STM32F303x8 up to 6 OC channels can be configured</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

### **HAL\_TIM\_ConfigClockSource**

|                             |                                                                                                                                                                                                                                    |
|-----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>        | <b>HAL_StatusTypeDef HAL_TIM_ConfigClockSource<br/>(TIM_HandleTypeDef * htim, TIM_ClockConfigTypeDef *<br/>sClockSourceConfig)</b>                                                                                                 |
| <b>Function description</b> | Configures the clock source to be used.                                                                                                                                                                                            |
| <b>Parameters</b>           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>sClockSourceConfig:</b> pointer to a TIM_ClockConfigTypeDef structure that contains the clock source information for the TIM peripheral.</li> </ul> |
| <b>Return values</b>        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                             |

**HAL\_TIM\_ConfigTI1Input**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_ConfigTI1Input<br/>(TIM_HandleTypeDef * htim, uint32_t TI1_Selection)</b>                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Selects the signal connected to the TI1 input: direct from CH1_input or a XOR combination between CH1_input, CH2_input & CH3_input.                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle.</li> <li>• <b>TI1_Selection:</b> Indicate whether or not channel 1 is connected to the output of a XOR gate. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <b>TIM_TI1SELECTION_CH1:</b> The TIMx_CH1 pin is connected to TI1 input</li> <li>– <b>TIM_TI1SELECTION_XORCOMBINATION:</b> The TIMx_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                       |

**HAL\_TIM\_SlaveConfigSynchronization**

|                      |                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_SlaveConfigSynchronization<br/>(TIM_HandleTypeDef * htim, TIM_SlaveConfigTypeDef * sSlaveConfig)</b>                                                                                                                                                                                                                            |
| Function description | Configures the TIM in Slave mode.                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle.</li> <li>• <b>sSlaveConfig:</b> pointer to a TIM_SlaveConfigTypeDef structure that contains the selected trigger (internal trigger input, filtered timer input or external trigger input) and the ) and the Slave mode (Disable, Reset, Gated, Trigger, External clock mode 1).</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                       |

**HAL\_TIM\_SlaveConfigSynchronization\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef<br/>HAL_TIM_SlaveConfigSynchronization_IT<br/>(TIM_HandleTypeDef * htim, TIM_SlaveConfigTypeDef * sSlaveConfig)</b>                                                                                                                                                                                                                     |
| Function description | Configures the TIM in Slave mode in interrupt mode.                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle.</li> <li>• <b>sSlaveConfig:</b> pointer to a TIM_SlaveConfigTypeDef structure that contains the selected trigger (internal trigger input, filtered timer input or external trigger input) and the ) and the Slave mode (Disable, Reset, Gated, Trigger, External clock mode 1).</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                       |

**HAL\_TIM\_DMABurst\_WriteStart**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_DMABurst_WriteStart<br/>(TIM_HandleTypeDef * htim, uint32_t BurstBaseAddress,<br/>uint32_t BurstRequestSrc, uint32_t * BurstBuffer, uint32_t<br/>BurstLength)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Configure the DMA Burst to transfer Data from the memory to the TIM peripheral.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM handle</li><li>• <b>BurstBaseAddress:</b> TIM Base address from where the DMA will start the Data write This parameter can be one of the following values:<ul style="list-style-type: none"><li>- TIM_DMABASE_CR1</li><li>- TIM_DMABASE_CR2</li><li>- TIM_DMABASE_SMCR</li><li>- TIM_DMABASE_DIER</li><li>- TIM_DMABASE_SR</li><li>- TIM_DMABASE_EGR</li><li>- TIM_DMABASE_CCMR1</li><li>- TIM_DMABASE_CCMR2</li><li>- TIM_DMABASE_CCER</li><li>- TIM_DMABASE_CNT</li><li>- TIM_DMABASE_PSC</li><li>- TIM_DMABASE_ARR</li><li>- TIM_DMABASE_RCR</li><li>- TIM_DMABASE_CCR1</li><li>- TIM_DMABASE_CCR2</li><li>- TIM_DMABASE_CCR3</li><li>- TIM_DMABASE_CCR4</li><li>- TIM_DMABASE_BDTR</li><li>- TIM_DMABASE_DCR</li></ul></li><li>• <b>BurstRequestSrc:</b> TIM DMA Request sources This parameter can be one of the following values:<ul style="list-style-type: none"><li>- TIM_DMA_UPDATE: TIM update Interrupt source</li><li>- TIM_DMA_CC1: TIM Capture Compare 1 DMA source</li><li>- TIM_DMA_CC2: TIM Capture Compare 2 DMA source</li><li>- TIM_DMA_CC3: TIM Capture Compare 3 DMA source</li><li>- TIM_DMA_CC4: TIM Capture Compare 4 DMA source</li><li>- TIM_DMA_COM: TIM Commutation DMA source</li><li>- TIM_DMA_TRIGGER: TIM Trigger DMA source</li></ul></li><li>• <b>BurstBuffer:</b> The Buffer address.</li><li>• <b>BurstLength:</b> DMA Burst length. This parameter can be one value between: TIM_DMABURSTLENGTH_1TRANSFER and TIM_DMABURSTLENGTH_18TRANSFERS.</li><li>• <b>HAL:</b> status</li></ul> |
| Return values        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

**HAL\_TIM\_DMABurst\_MultiWriteStart**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_TIM_DMABurst_MultiWriteStart<br/>(TIM_HandleTypeDef * htim, uint32_t BurstBaseAddress,<br/>uint32_t BurstRequestSrc, uint32_t * BurstBuffer, uint32_t<br/>BurstLength, uint32_t DataLength)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Configure the DMA Burst to transfer multiple Data from the memory to the TIM peripheral.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>BurstBaseAddress:</b> TIM Base address from where the DMA will start the Data write This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>TIM_DMABASE_CR1</code></li> <li>- <code>TIM_DMABASE_CR2</code></li> <li>- <code>TIM_DMABASE_SMCR</code></li> <li>- <code>TIM_DMABASE_DIER</code></li> <li>- <code>TIM_DMABASE_SR</code></li> <li>- <code>TIM_DMABASE_EGR</code></li> <li>- <code>TIM_DMABASE_CCMR1</code></li> <li>- <code>TIM_DMABASE_CCMR2</code></li> <li>- <code>TIM_DMABASE_CCER</code></li> <li>- <code>TIM_DMABASE_CNT</code></li> <li>- <code>TIM_DMABASE_PSC</code></li> <li>- <code>TIM_DMABASE_ARR</code></li> <li>- <code>TIM_DMABASE_RCR</code></li> <li>- <code>TIM_DMABASE_CCR1</code></li> <li>- <code>TIM_DMABASE_CCR2</code></li> <li>- <code>TIM_DMABASE_CCR3</code></li> <li>- <code>TIM_DMABASE_CCR4</code></li> <li>- <code>TIM_DMABASE_BDTR</code></li> <li>- <code>TIM_DMABASE_DCR</code></li> </ul> </li> <li>• <b>BurstRequestSrc:</b> TIM DMA Request sources This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>TIM_DMA_UPDATE</code>: TIM update Interrupt source</li> <li>- <code>TIM_DMA_CC1</code>: TIM Capture Compare 1 DMA source</li> <li>- <code>TIM_DMA_CC2</code>: TIM Capture Compare 2 DMA source</li> <li>- <code>TIM_DMA_CC3</code>: TIM Capture Compare 3 DMA source</li> <li>- <code>TIM_DMA_CC4</code>: TIM Capture Compare 4 DMA source</li> <li>- <code>TIM_DMA_COM</code>: TIM Commutation DMA source</li> <li>- <code>TIM_DMA_TRIGGER</code>: TIM Trigger DMA source</li> </ul> </li> <li>• <b>BurstBuffer:</b> The Buffer address.</li> <li>• <b>BurstLength:</b> DMA Burst length. This parameter can be one value between: <code>TIM_DMABURSTLENGTH_1TRANSFER</code> and <code>TIM_DMABURSTLENGTH_18TRANSFERS</code>.</li> <li>• <b>DataLength:</b> Data length. This parameter can be one value between 1 and <code>0xFFFF</code>.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

**HAL\_TIM\_DMABurst\_WriteStop**

|                      |                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_DMABurst_WriteStop<br/>(TIM_HandleTypeDef * htim, uint32_t BurstRequestSrc)</b>                                      |
| Function description | Stops the TIM DMA Burst mode.                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>BurstRequestSrc:</b> TIM DMA Request sources to disable</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                            |

**HAL\_TIM\_DMABurst\_ReadStart**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_DMABurst_ReadStart<br/>(TIM_HandleTypeDef * htim, uint32_t BurstBaseAddress,<br/>uint32_t BurstRequestSrc, uint32_t * BurstBuffer, uint32_t<br/>BurstLength)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Configure the DMA Burst to transfer Data from the TIM peripheral to the memory.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>BurstBaseAddress:</b> TIM Base address from where the DMA will starts the Data read This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_DMABASE_CR1</li> <li>- TIM_DMABASE_CR2</li> <li>- TIM_DMABASE_SMCR</li> <li>- TIM_DMABASE_DIER</li> <li>- TIM_DMABASE_SR</li> <li>- TIM_DMABASE_EGR</li> <li>- TIM_DMABASE_CCMR1</li> <li>- TIM_DMABASE_CCMR2</li> <li>- TIM_DMABASE_CCER</li> <li>- TIM_DMABASE_CNT</li> <li>- TIM_DMABASE_PSC</li> <li>- TIM_DMABASE_ARR</li> <li>- TIM_DMABASE_RCR</li> <li>- TIM_DMABASE_CCR1</li> <li>- TIM_DMABASE_CCR2</li> <li>- TIM_DMABASE_CCR3</li> <li>- TIM_DMABASE_CCR4</li> <li>- TIM_DMABASE_BDTR</li> <li>- TIM_DMABASE_DCR</li> </ul> </li> <li>• <b>BurstRequestSrc:</b> TIM DMA Request sources This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_DMA_UPDATE: TIM update Interrupt source</li> <li>- TIM_DMA_CC1: TIM Capture Compare 1 DMA source</li> <li>- TIM_DMA_CC2: TIM Capture Compare 2 DMA source</li> <li>- TIM_DMA_CC3: TIM Capture Compare 3 DMA source</li> <li>- TIM_DMA_CC4: TIM Capture Compare 4 DMA source</li> <li>- TIM_DMA_COM: TIM Commutation DMA source</li> <li>- TIM_DMA_TRIGGER: TIM Trigger DMA source</li> </ul> </li> <li>• <b>BurstBuffer:</b> The Buffer address.</li> <li>• <b>BurstLength:</b> DMA Burst length. This parameter can be one</li> </ul> |

value between: TIM\_DMABURSTLENGTH\_1TRANSFER and  
TIM\_DMABURSTLENGTH\_18TRANSFERS.

|                                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                          | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>HAL_TIM_DMABurst_MultiReadStart</b> |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function name                          | <b>HAL_StatusTypeDef HAL_DMABurst_MultiReadStart<br/>(TIM_HandleTypeDef * htim, uint32_t BurstBaseAddress,<br/>uint32_t BurstRequestSrc, uint32_t * BurstBuffer, uint32_t<br/>BurstLength, uint32_t DataLength)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description                   | Configure the DMA Burst to transfer multiple Data from the TIM peripheral to the memory.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                             | <ul style="list-style-type: none"> <li><b>htim:</b> TIM handle</li> <li><b>BurstBaseAddress:</b> TIM Base address from where the DMA will starts the Data read This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_DMABASE_CR1</li> <li>- TIM_DMABASE_CR2</li> <li>- TIM_DMABASE_SMCR</li> <li>- TIM_DMABASE_DIER</li> <li>- TIM_DMABASE_SR</li> <li>- TIM_DMABASE_EGR</li> <li>- TIM_DMABASE_CCMR1</li> <li>- TIM_DMABASE_CCMR2</li> <li>- TIM_DMABASE_CCER</li> <li>- TIM_DMABASE_CNT</li> <li>- TIM_DMABASE_PSC</li> <li>- TIM_DMABASE_ARR</li> <li>- TIM_DMABASE_RCR</li> <li>- TIM_DMABASE_CCR1</li> <li>- TIM_DMABASE_CCR2</li> <li>- TIM_DMABASE_CCR3</li> <li>- TIM_DMABASE_CCR4</li> <li>- TIM_DMABASE_BDTR</li> <li>- TIM_DMABASE_DCR</li> </ul> </li> <li><b>BurstRequestSrc:</b> TIM DMA Request sources This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_DMA_UPDATE: TIM update Interrupt source</li> <li>- TIM_DMA_CC1: TIM Capture Compare 1 DMA source</li> <li>- TIM_DMA_CC2: TIM Capture Compare 2 DMA source</li> <li>- TIM_DMA_CC3: TIM Capture Compare 3 DMA source</li> <li>- TIM_DMA_CC4: TIM Capture Compare 4 DMA source</li> <li>- TIM_DMA_COM: TIM Commutation DMA source</li> <li>- TIM_DMA_TRIGGER: TIM Trigger DMA source</li> </ul> </li> <li><b>BurstBuffer:</b> The Buffer address.</li> <li><b>BurstLength:</b> DMA Burst length. This parameter can be one value between: TIM_DMABURSTLENGTH_1TRANSFER and TIM_DMABURSTLENGTH_18TRANSFERS.</li> <li><b>DataLength:</b> Data length. This parameter can be one value between 1 and 0xFFFF.</li> </ul> |
| Return values                          | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

**HAL\_TIM\_DMABurst\_ReadStop**

|                      |                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_DMABurst_ReadStop<br/>(TIM_HandleTypeDef * htim, uint32_t BurstRequestSrc)</b>                                        |
| Function description | Stop the DMA burst reading.                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>BurstRequestSrc:</b> TIM DMA Request sources to disable.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                             |

**HAL\_TIM\_GenerateEvent**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIM_GenerateEvent<br/>(TIM_HandleTypeDef * htim, uint32_t EventSource)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Generate a software event.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>EventSource:</b> specifies the event source. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_EVENTSOURCE_UPDATE: Timer update Event source</li> <li>- TIM_EVENTSOURCE_CC1: Timer Capture Compare 1 Event source</li> <li>- TIM_EVENTSOURCE_CC2: Timer Capture Compare 2 Event source</li> <li>- TIM_EVENTSOURCE_CC3: Timer Capture Compare 3 Event source</li> <li>- TIM_EVENTSOURCE_CC4: Timer Capture Compare 4 Event source</li> <li>- TIM_EVENTSOURCE_COM: Timer COM event source</li> <li>- TIM_EVENTSOURCE_TRIGGER: Timer Trigger Event source</li> <li>- TIM_EVENTSOURCE_BREAK: Timer Break event source</li> <li>- TIM_EVENTSOURCE_BREAK2: Timer Break2 event source</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• TIM_EVENTSOURCE_BREAK2 isn't relevant for STM32F37xx and STM32F38xx devices</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

**HAL\_TIM\_ReadCapturedValue**

|                      |                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_TIM_ReadCapturedValue (TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                             |
| Function description | Read the captured value from Capture Compare unit.                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle.</li> <li>• <b>Channel:</b> TIM Channels to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>- TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>- TIM_CHANNEL_3: TIM Channel 3 selected</li> </ul> </li> </ul> |

- TIM\_CHANNEL\_4: TIM Channel 4 selected

Return values

- **Captured:** value

### **HAL\_TIM\_PeriodElapsedCallback**

Function name **void HAL\_TIM\_PeriodElapsedCallback (TIM\_HandleTypeDef \* htim)**

Function description Period elapsed callback in non blocking mode.

Parameters

- **htim:** TIM handle

Return values

- **None**

### **HAL\_TIM\_OC\_DelayElapsedCallback**

Function name **void HAL\_TIM\_OC\_DelayElapsedCallback (TIM\_HandleTypeDef \* htim)**

Function description Output Compare callback in non blocking mode.

Parameters

- **htim:** TIM OC handle

Return values

- **None**

### **HAL\_TIM\_IC\_CaptureCallback**

Function name **void HAL\_TIM\_IC\_CaptureCallback (TIM\_HandleTypeDef \* htim)**

Function description Input Capture callback in non blocking mode.

Parameters

- **htim:** TIM IC handle

Return values

- **None**

### **HAL\_TIM\_PWM\_PulseFinishedCallback**

Function name **void HAL\_TIM\_PWM\_PulseFinishedCallback (TIM\_HandleTypeDef \* htim)**

Function description PWM Pulse finished callback in non blocking mode.

Parameters

- **htim:** TIM handle

Return values

- **None**

### **HAL\_TIM\_TriggerCallback**

Function name **void HAL\_TIM\_TriggerCallback (TIM\_HandleTypeDef \* htim)**

Function description Hall Trigger detection callback in non blocking mode.

Parameters

- **htim:** TIM handle

Return values

- **None**

**HAL\_TIM\_ErrorCallback**

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>void HAL_TIM_ErrorCallback (TIM_HandleTypeDef * htim)</b>              |
| Function description | Timer error callback in non blocking mode.                                |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>             |

**HAL\_TIM\_Base\_GetState**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>HAL_TIM_StateTypeDef HAL_TIM_Base_GetState (TIM_HandleTypeDef * htim)</b>   |
| Function description | Return the TIM Base state.                                                     |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM Base handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> state</li></ul>            |

**HAL\_TIM\_OC\_GetState**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b>HAL_TIM_StateTypeDef HAL_TIM_OC_GetState (TIM_HandleTypeDef * htim)</b>              |
| Function description | Return the TIM OC state.                                                                |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM Ouput Compare handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> state</li></ul>                     |

**HAL\_TIM\_PWM\_GetState**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>HAL_TIM_StateTypeDef HAL_TIM_PWM_GetState (TIM_HandleTypeDef * htim)</b> |
| Function description | Return the TIM PWM state.                                                   |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM handle</li></ul>   |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> state</li></ul>         |

**HAL\_TIM\_IC\_GetState**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>HAL_TIM_StateTypeDef HAL_TIM_IC_GetState (TIM_HandleTypeDef * htim)</b>   |
| Function description | Return the TIM Input Capture state.                                          |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htim:</b> TIM IC handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> state</li></ul>          |

**HAL\_TIM\_OnePulse\_GetState**

|                      |                                                                                      |
|----------------------|--------------------------------------------------------------------------------------|
| Function name        | <b>HAL_TIM_StateTypeDef HAL_TIM_OnePulse_GetState<br/>(TIM_HandleTypeDef * htim)</b> |
| Function description | Return the TIM One Pulse Mode state.                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM OPM handle</li> </ul>      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                |

**HAL\_TIM\_Encoder\_GetState**

|                      |                                                                                     |
|----------------------|-------------------------------------------------------------------------------------|
| Function name        | <b>HAL_TIM_StateTypeDef HAL_TIM_Encoder_GetState<br/>(TIM_HandleTypeDef * htim)</b> |
| Function description | Return the TIM Encoder Mode state.                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Encoder handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>               |

**TIM\_Base\_SetConfig**

|                      |                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void TIM_Base_SetConfig (TIM_TypeDef * TIMx,<br/>TIM_Base_InitTypeDef * Structure)</b>                                                     |
| Function description | Time Base configuration.                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> TIM peripheral</li> <li>• <b>Structure:</b> TIM Base configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                               |

**TIM\_TI1\_SetConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void TIM_TI1_SetConfig (TIM_TypeDef * TIMx, uint32_t<br/>TIM_ICPolarity, uint32_t TIM_ICSelection, uint32_t<br/>TIM_ICFilter)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Configure the TI1 as Input.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> to select the TIM peripheral.</li> <li>• <b>TIM_ICPolarity:</b> The Input Polarity. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_ICPOLARITY_RISING</li> <li>– TIM_ICPOLARITY_FALLING</li> <li>– TIM_ICPOLARITY_BOTHEDGE</li> </ul> </li> <li>• <b>TIM_ICSelection:</b> specifies the input to be used. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_ICSELECTION_DIRECTTI: TIM Input 1 is selected to be connected to IC1.</li> <li>– TIM_ICSELECTION_INDIRECTTI: TIM Input 1 is selected to be connected to IC2.</li> <li>– TIM_ICSELECTION_TRC: TIM Input 1 is selected to be connected to TRC.</li> </ul> </li> <li>• <b>TIM_ICFilter:</b> Specifies the Input Capture Filter. This parameter must be a value between 0x00 and 0x0F.</li> </ul> |

|               |                                                                                                                                                                                                                                                               |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                 |
| Notes         | <ul style="list-style-type: none"> <li>TIM_ICFilter and TIM_ICPolarity are not used in INDIRECT mode as TI2FP1 (on channel2 path) is used as the input signal. Therefore CCMR1 must be protected against uninitialized filter and polarity values.</li> </ul> |

### **TIM\_OC1\_SetConfig**

|                      |                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void TIM_OC1_SetConfig (TIM_TypeDef * TIMx,<br/>TIM_OC_InitTypeDef * OC_Config)</b>                                                                   |
| Function description | Time Ouput Compare 1 configuration.                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>TIMx:</b> to select the TIM peripheral</li> <li><b>OC_Config:</b> The ouput configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                            |

### **TIM\_OC2\_SetConfig**

|                      |                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void TIM_OC2_SetConfig (TIM_TypeDef * TIMx,<br/>TIM_OC_InitTypeDef * OC_Config)</b>                                                                   |
| Function description | Time Ouput Compare 2 configuration.                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>TIMx:</b> to select the TIM peripheral</li> <li><b>OC_Config:</b> The ouput configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                            |

### **TIM\_OC3\_SetConfig**

|                      |                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void TIM_OC3_SetConfig (TIM_TypeDef * TIMx,<br/>TIM_OC_InitTypeDef * OC_Config)</b>                                                                   |
| Function description | Time Ouput Compare 3 configuration.                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>TIMx:</b> to select the TIM peripheral</li> <li><b>OC_Config:</b> The ouput configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                            |

### **TIM\_OC4\_SetConfig**

|                      |                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void TIM_OC4_SetConfig (TIM_TypeDef * TIMx,<br/>TIM_OC_InitTypeDef * OC_Config)</b>                                                                   |
| Function description | Time Ouput Compare 4 configuration.                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>TIMx:</b> to select the TIM peripheral</li> <li><b>OC_Config:</b> The ouput configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                            |

**TIM\_ETR\_SetConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void TIM_ETR_SetConfig (TIM_TypeDef * TIMx, uint32_t TIM_ExtTRGPrescaler, uint32_t TIM_ExtTRGPolarity, uint32_t ExtTRGFilter)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Configures the TIMx External Trigger (ETR).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> to select the TIM peripheral</li> <li>• <b>TIM_ExtTRGPrescaler:</b> The external Trigger Prescaler. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_ETRPRESCALER_DIV1 : ETRP Prescaler OFF.</li> <li>– TIM_ETRPRESCALER_DIV2 : ETRP frequency divided by 2.</li> <li>– TIM_ETRPRESCALER_DIV4 : ETRP frequency divided by 4.</li> <li>– TIM_ETRPRESCALER_DIV8 : ETRP frequency divided by 8.</li> </ul> </li> <li>• <b>TIM_ExtTRGPolarity:</b> The external Trigger Polarity. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_ETRPOLARITY_INVERTED : active low or falling edge active.</li> <li>– TIM_ETRPOLARITY_NONINVERTED : active high or rising edge active.</li> </ul> </li> <li>• <b>ExtTRGFilter:</b> External Trigger Filter. This parameter must be a value between 0x00 and 0x0F</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

**TIM\_DMADelayPulseCplt**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b>void TIM_DMADelayPulseCplt (DMA_HandleTypeDef * hdma)</b>                            |
| Function description | TIM DMA Delay Pulse complete callback.                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to DMA handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |

**TIM\_DMSError**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b>void TIM_DMSError (DMA_HandleTypeDef * hdma)</b>                                     |
| Function description | TIM DMA error callback.                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to DMA handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |

**TIM\_DMACaptureCplt**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b>void TIM_DMACaptureCplt (DMA_HandleTypeDef * hdma)</b>                               |
| Function description | TIM DMA Capture complete callback.                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to DMA handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |

**TIM\_CCxChannelCmd**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void TIM_CCxChannelCmd (TIM_TypeDef * TIMx, uint32_t Channel, uint32_t ChannelState)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Enables or disables the TIM Capture Compare Channel x.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> to select the TIM peripheral</li> <li>• <b>Channel:</b> specifies the TIM Channel This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_CHANNEL_1: TIM Channel 1</li> <li>- TIM_CHANNEL_2: TIM Channel 2</li> <li>- TIM_CHANNEL_3: TIM Channel 3</li> <li>- TIM_CHANNEL_4: TIM Channel 4</li> </ul> </li> <li>• <b>ChannelState:</b> specifies the TIM Channel CCxE bit new state. This parameter can be: TIM_CCx_ENABLE or TIM_CCx_Disable.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

## 51.3 TIM Firmware driver defines

### 51.3.1 TIM

***TIM Automatic Output Enable***`TIM_AUTOMATICOUTPUT_ENABLE``TIM_AUTOMATICOUTPUT_DISABLE`***TIM Auto-Reload Preload***`TIM_AUTORELOAD_PRELOAD_DISABLE` TIMx\_ARR register is not buffered`TIM_AUTORELOAD_PRELOAD_ENABLE` TIMx\_ARR register is buffered***TIM Break Input Enable Disable***`TIM_BREAK_ENABLE``TIM_BREAK_DISABLE`***TIM Break Input Polarity***`TIM_BREAKPOLARITY_LOW``TIM_BREAKPOLARITY_HIGH`***TIM Capture/Compare Channel State***`TIM_CCx_ENABLE``TIM_CCx_DISABLE``TIM_CCxN_ENABLE``TIM_CCxN_DISABLE`***TIM Clear Input Polarity***`TIM_CLEARINPUTPOLARITY_INVERTED` Polarity for ETRx pin`TIM_CLEARINPUTPOLARITY_NONINVERTED` Polarity for ETRx pin

***TIM Clear Input Prescaler***

|                                           |                                                                        |
|-------------------------------------------|------------------------------------------------------------------------|
| <code>TIM_CLEARINPUTPRESCALER_DIV1</code> | No prescaler is used                                                   |
| <code>TIM_CLEARINPUTPRESCALER_DIV2</code> | Prescaler for External ETR pin: Capture performed once every 2 events. |
| <code>TIM_CLEARINPUTPRESCALER_DIV4</code> | Prescaler for External ETR pin: Capture performed once every 4 events. |
| <code>TIM_CLEARINPUTPRESCALER_DIV8</code> | Prescaler for External ETR pin: Capture performed once every 8 events. |

***TIM Clock Division***

|                                     |
|-------------------------------------|
| <code>TIM_CLOCKDIVISION_DIV1</code> |
| <code>TIM_CLOCKDIVISION_DIV2</code> |
| <code>TIM_CLOCKDIVISION_DIV4</code> |

***TIM Clock Polarity***

|                                            |                                 |
|--------------------------------------------|---------------------------------|
| <code>TIM_CLOCKPOLARITY_INVERTED</code>    | Polarity for ETRx clock sources |
| <code>TIM_CLOCKPOLARITY_NONINVERTED</code> | Polarity for ETRx clock sources |
| <code>TIM_CLOCKPOLARITY_RISING</code>      | Polarity for TIx clock sources  |
| <code>TIM_CLOCKPOLARITY_FALLING</code>     | Polarity for TIx clock sources  |
| <code>TIM_CLOCKPOLARITY_BOTHEDGE</code>    | Polarity for TIx clock sources  |

***TIM Clock Prescaler***

|                                      |                                                                          |
|--------------------------------------|--------------------------------------------------------------------------|
| <code>TIM_CLOCKPRESCALER_DIV1</code> | No prescaler is used                                                     |
| <code>TIM_CLOCKPRESCALER_DIV2</code> | Prescaler for External ETR Clock: Capture performed once every 2 events. |
| <code>TIM_CLOCKPRESCALER_DIV4</code> | Prescaler for External ETR Clock: Capture performed once every 4 events. |
| <code>TIM_CLOCKPRESCALER_DIV8</code> | Prescaler for External ETR Clock: Capture performed once every 8 events. |

***TIM Clock Source***

|                                       |
|---------------------------------------|
| <code>TIM_CLOCKSOURCE_ETRMODE2</code> |
| <code>TIM_CLOCKSOURCE_INTERNAL</code> |
| <code>TIM_CLOCKSOURCE_ITR0</code>     |
| <code>TIM_CLOCKSOURCE_ITR1</code>     |
| <code>TIM_CLOCKSOURCE_ITR2</code>     |
| <code>TIM_CLOCKSOURCE_ITR3</code>     |
| <code>TIM_CLOCKSOURCE_TI1ED</code>    |
| <code>TIM_CLOCKSOURCE_TI1</code>      |
| <code>TIM_CLOCKSOURCE_TI2</code>      |
| <code>TIM_CLOCKSOURCE_ETRMODE1</code> |

***TIM Commutation Source***

TIM\_COMMUTATION\_TRGI  
TIM\_COMMUTATION\_SOFTWARE

***TIM Counter Mode***

TIM\_COUNTERMODE\_UP  
TIM\_COUNTERMODE\_DOWN  
TIM\_COUNTERMODE\_CENTERALIGNED1  
TIM\_COUNTERMODE\_CENTERALIGNED2  
TIM\_COUNTERMODE\_CENTERALIGNED3

***TIMEx DMA Base Address***

TIM\_DMABASE\_CR1  
TIM\_DMABASE\_CR2  
TIM\_DMABASE\_SMCR  
TIM\_DMABASE\_DIER  
TIM\_DMABASE\_SR  
TIM\_DMABASE\_EGR  
TIM\_DMABASE\_CCMR1  
TIM\_DMABASE\_CCMR2  
TIM\_DMABASE\_CCER  
TIM\_DMABASE\_CNT  
TIM\_DMABASE\_PSC  
TIM\_DMABASE\_ARR  
TIM\_DMABASE\_RCR  
TIM\_DMABASE\_CCR1  
TIM\_DMABASE\_CCR2  
TIM\_DMABASE\_CCR3  
TIM\_DMABASE\_CCR4  
TIM\_DMABASE\_BDTR  
TIM\_DMABASE\_DCR  
TIM\_DMABASE\_CCMR3  
TIM\_DMABASE\_CCR5  
TIM\_DMABASE\_CCR6  
TIM\_DMABASE\_OR

***TIM DMA Burst Length***

TIM\_DMABURSTLENGTH\_1TRANSFER  
TIM\_DMABURSTLENGTH\_2TRANSFERS

TIM\_DMABURSTLENGTH\_3TRANSFERS  
TIM\_DMABURSTLENGTH\_4TRANSFERS  
TIM\_DMABURSTLENGTH\_5TRANSFERS  
TIM\_DMABURSTLENGTH\_6TRANSFERS  
TIM\_DMABURSTLENGTH\_7TRANSFERS  
TIM\_DMABURSTLENGTH\_8TRANSFERS  
TIM\_DMABURSTLENGTH\_9TRANSFERS  
TIM\_DMABURSTLENGTH\_10TRANSFERS  
TIM\_DMABURSTLENGTH\_11TRANSFERS  
TIM\_DMABURSTLENGTH\_12TRANSFERS  
TIM\_DMABURSTLENGTH\_13TRANSFERS  
TIM\_DMABURSTLENGTH\_14TRANSFERS  
TIM\_DMABURSTLENGTH\_15TRANSFERS  
TIM\_DMABURSTLENGTH\_16TRANSFERS  
TIM\_DMABURSTLENGTH\_17TRANSFERS  
TIM\_DMABURSTLENGTH\_18TRANSFERS

***TIM DMA Handle Index***

|                        |                                                                 |
|------------------------|-----------------------------------------------------------------|
| TIM_DMA_ID_UPDATE      | Index of the DMA handle used for Update DMA requests            |
| TIM_DMA_ID_CC1         | Index of the DMA handle used for Capture/Compare 1 DMA requests |
| TIM_DMA_ID_CC2         | Index of the DMA handle used for Capture/Compare 2 DMA requests |
| TIM_DMA_ID_CC3         | Index of the DMA handle used for Capture/Compare 3 DMA requests |
| TIM_DMA_ID_CC4         | Index of the DMA handle used for Capture/Compare 4 DMA requests |
| TIM_DMA_ID_COMMUTATION | Index of the DMA handle used for Commutation DMA requests       |
| TIM_DMA_ID_TRIGGER     | Index of the DMA handle used for Trigger DMA requests           |

***TIM DMA Sources***

TIM\_DMA\_UPDATE  
TIM\_DMA\_CC1  
TIM\_DMA\_CC2  
TIM\_DMA\_CC3  
TIM\_DMA\_CC4  
TIM\_DMA\_COM  
TIM\_DMA\_TRIGGER

***TIM Encoder Mode***

`TIM_ENCODERMODE_TI1`  
`TIM_ENCODERMODE_TI2`  
`TIM_ENCODERMODE_TI12`

***TIM ETR Polarity***

`TIM_ETRPOLARITY_INVERTED`      Polarity for ETR source  
`TIM_ETRPOLARITY_NONINVERTED`    Polarity for ETR source

***TIM ETR Prescaler***

`TIM_ETRPRESCALER_DIV1`    No prescaler is used  
`TIM_ETRPRESCALER_DIV2`    ETR input source is divided by 2U  
`TIM_ETRPRESCALER_DIV4`    ETR input source is divided by 4U  
`TIM_ETRPRESCALER_DIV8`    ETR input source is divided by 8U

***TIMEx Event Source***

|                                      |                                                                   |
|--------------------------------------|-------------------------------------------------------------------|
| <code>TIM_EVENTSOURCE_UPDATE</code>  | Reinitialize the counter and generates an update of the registers |
| <code>TIM_EVENTSOURCE_CC1</code>     | A capture/compare event is generated on channel 1U                |
| <code>TIM_EVENTSOURCE_CC2</code>     | A capture/compare event is generated on channel 2U                |
| <code>TIM_EVENTSOURCE_CC3</code>     | A capture/compare event is generated on channel 3U                |
| <code>TIM_EVENTSOURCE_CC4</code>     | A capture/compare event is generated on channel 4U                |
| <code>TIM_EVENTSOURCE_COM</code>     | A commutation event is generated                                  |
| <code>TIM_EVENTSOURCE_TRIGGER</code> | A trigger event is generated                                      |
| <code>TIM_EVENTSOURCE_BREAK</code>   | A break event is generated                                        |
| <code>TIM_EVENTSOURCE_BREAK2</code>  | A break 2 event is generated                                      |

***TIM Exported Macros***

|                                          |                                                                                                                                                                                                                                                                                                        |
|------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>_HAL_TIM_RESET_HANDLE_STATE</code> | <b>Description:</b><br><br><ul style="list-style-type: none"> <li>• Reset TIM handle state.</li> </ul> <b>Parameters:</b><br><ul style="list-style-type: none"> <li>• <code>_HANDLE_</code>: TIM handle.</li> </ul> <b>Return value:</b><br><ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <code>_HAL_TIM_ENABLE</code>             | <b>Description:</b><br><br><ul style="list-style-type: none"> <li>• Enable the TIM peripheral.</li> </ul> <b>Parameters:</b><br><ul style="list-style-type: none"> <li>• <code>_HANDLE_</code>: TIM handle</li> </ul> <b>Return value:</b><br><ul style="list-style-type: none"> <li>• None</li> </ul> |

|                                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>__HAL_TIM_MOE_ENABLE</code>                  | <p><b>Description:</b></p> <ul style="list-style-type: none"> <li>Enable the TIM main Output.</li> </ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"> <li><code>__HANDLE__</code>: TIM handle</li> </ul> <p><b>Return value:</b></p> <ul style="list-style-type: none"> <li>None</li> </ul>                                                                                                                                                                                         |
| <code>__HAL_TIM_DISABLE</code>                     | <p><b>Description:</b></p> <ul style="list-style-type: none"> <li>Disable the TIM peripheral.</li> </ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"> <li><code>__HANDLE__</code>: TIM handle</li> </ul> <p><b>Return value:</b></p> <ul style="list-style-type: none"> <li>None</li> </ul>                                                                                                                                                                                         |
| <code>__HAL_TIM_MOE_DISABLE</code>                 | <p><b>Description:</b></p> <ul style="list-style-type: none"> <li>Disable the TIM main Output.</li> </ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"> <li><code>__HANDLE__</code>: TIM handle</li> </ul> <p><b>Return value:</b></p> <ul style="list-style-type: none"> <li>None</li> </ul> <p><b>Notes:</b></p> <ul style="list-style-type: none"> <li>The Main Output Enable of a timer instance is disabled only if all the CCx and CCxN channels have been disabled</li> </ul> |
| <code>__HAL_TIM_MOE_DISABLE_UNCONDITIONALLY</code> | <p><b>Description:</b></p> <ul style="list-style-type: none"> <li>Disable the TIM main Output.</li> </ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"> <li><code>__HANDLE__</code>: TIM handle</li> </ul> <p><b>Return value:</b></p> <ul style="list-style-type: none"> <li>None</li> </ul> <p><b>Notes:</b></p> <ul style="list-style-type: none"> <li>The Main Output Enable of a timer instance is disabled unconditionally</li> </ul>                                          |
| <code>__HAL_TIM_ENABLE_IT</code>                   | <p><b>Description:</b></p> <ul style="list-style-type: none"> <li>Enables the specified TIM interrupt.</li> </ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"> <li><code>__HANDLE__</code>: specifies the TIM Handle.</li> <li><code>__INTERRUPT__</code>: specifies the TIM interrupt source to enable. This parameter can be one of the following values:</li> </ul>                                                                                                              |

- TIM\_IT\_UPDATE: Update interrupt
- TIM\_IT\_CC1: Capture/Compare 1 interrupt
- TIM\_IT\_CC2: Capture/Compare 2 interrupt
- TIM\_IT\_CC3: Capture/Compare 3 interrupt
- TIM\_IT\_CC4: Capture/Compare 4 interrupt
- TIM\_IT\_COM: Commutation interrupt
- TIM\_IT\_TRIGGER: Trigger interrupt
- TIM\_IT\_BREAK: Break interrupt

**Return value:**

- None

`__HAL_TIM_DISABLE_IT`

**Description:**

- Disables the specified TIM interrupt.

**Parameters:**

- `__HANDLE__`: specifies the TIM Handle.
- `__INTERRUPT__`: specifies the TIM interrupt source to disable. This parameter can be one of the following values:
  - TIM\_IT\_UPDATE: Update interrupt
  - TIM\_IT\_CC1: Capture/Compare 1 interrupt
  - TIM\_IT\_CC2: Capture/Compare 2 interrupt
  - TIM\_IT\_CC3: Capture/Compare 3 interrupt
  - TIM\_IT\_CC4: Capture/Compare 4 interrupt
  - TIM\_IT\_COM: Commutation interrupt
  - TIM\_IT\_TRIGGER: Trigger interrupt
  - TIM\_IT\_BREAK: Break interrupt

**Return value:**

- None

`__HAL_TIM_ENABLE_DMA`

**Description:**

- Enables the specified DMA request.

**Parameters:**

- `__HANDLE__`: specifies the TIM Handle.
- `__DMA__`: specifies the TIM DMA request to enable. This parameter can be one of the following values:
  - TIM\_DMA\_UPDATE: Update DMA request
  - TIM\_DMA\_CC1: Capture/Compare 1 DMA request
  - TIM\_DMA\_CC2: Capture/Compare 2 DMA request

- TIM\_DMA\_CC3: Capture/Compare 3 DMA request
- TIM\_DMA\_CC4: Capture/Compare 4 DMA request
- TIM\_DMA\_COM: Commutation DMA request
- TIM\_DMA\_TRIGGER: Trigger DMA request

**Return value:**

- None

`_HAL_TIM_DISABLE_DMA`

**Description:**

- Disables the specified DMA request.

**Parameters:**

- `_HANDLE_`: specifies the TIM Handle.
- `_DMA_`: specifies the TIM DMA request to disable. This parameter can be one of the following values:
  - TIM\_DMA\_UPDATE: Update DMA request
  - TIM\_DMA\_CC1: Capture/Compare 1 DMA request
  - TIM\_DMA\_CC2: Capture/Compare 2 DMA request
  - TIM\_DMA\_CC3: Capture/Compare 3 DMA request
  - TIM\_DMA\_CC4: Capture/Compare 4 DMA request
  - TIM\_DMA\_COM: Commutation DMA request
  - TIM\_DMA\_TRIGGER: Trigger DMA request

**Return value:**

- None

`_HAL_TIM_GET_FLAG`

**Description:**

- Checks whether the specified TIM interrupt flag is set or not.

**Parameters:**

- `_HANDLE_`: specifies the TIM Handle.
- `_FLAG_`: specifies the TIM interrupt flag to check. This parameter can be one of the following values:
  - TIM\_FLAG\_UPDATE: Update interrupt flag
  - TIM\_FLAG\_CC1: Capture/Compare 1 interrupt flag
  - TIM\_FLAG\_CC2: Capture/Compare 2 interrupt flag
  - TIM\_FLAG\_CC3: Capture/Compare 3 interrupt flag

- interrupt flag
- TIM\_FLAG\_CC4: Capture/Compare 4 interrupt flag
- TIM\_FLAG\_COM: Commutation interrupt flag
- TIM\_FLAG\_TRIGGER: Trigger interrupt flag
- TIM\_FLAG\_BREAK: Break interrupt flag
- TIM\_FLAG\_CC1OF: Capture/Compare 1 overcapture flag
- TIM\_FLAG\_CC2OF: Capture/Compare 2 overcapture flag
- TIM\_FLAG\_CC3OF: Capture/Compare 3 overcapture flag
- TIM\_FLAG\_CC4OF: Capture/Compare 4 overcapture flag

**Return value:**

- The new state of `__FLAG__` (TRUE or FALSE).

**\_HAL\_TIM\_CLEAR\_FLAG****Description:**

- Clears the specified TIM interrupt flag.

**Parameters:**

- `__HANDLE__`: specifies the TIM Handle.
- `__FLAG__`: specifies the TIM interrupt flag to clear. This parameter can be one of the following values:
  - TIM\_FLAG\_UPDATE: Update interrupt flag
  - TIM\_FLAG\_CC1: Capture/Compare 1 interrupt flag
  - TIM\_FLAG\_CC2: Capture/Compare 2 interrupt flag
  - TIM\_FLAG\_CC3: Capture/Compare 3 interrupt flag
  - TIM\_FLAG\_CC4: Capture/Compare 4 interrupt flag
  - TIM\_FLAG\_COM: Commutation interrupt flag
  - TIM\_FLAG\_TRIGGER: Trigger interrupt flag
  - TIM\_FLAG\_BREAK: Break interrupt flag
  - TIM\_FLAG\_CC1OF: Capture/Compare 1 overcapture flag
  - TIM\_FLAG\_CC2OF: Capture/Compare 2 overcapture flag
  - TIM\_FLAG\_CC3OF: Capture/Compare 3 overcapture flag
  - TIM\_FLAG\_CC4OF:

---

Capture/Compare 4 overcapture flag

**Return value:**

- The: new state of \_\_FLAG\_\_ (TRUE or FALSE).

`__HAL_TIM_GET_IT_SOURCE`

**Description:**

- Checks whether the specified TIM interrupt has occurred or not.

**Parameters:**

- \_\_HANDLE\_\_: TIM handle
- \_\_INTERRUPT\_\_: specifies the TIM interrupt source to check.

**Return value:**

- The: state of TIM\_IT (SET or RESET).

`__HAL_TIM_CLEAR_IT`

**Description:**

- Clear the TIM interrupt pending bits.

**Parameters:**

- \_\_HANDLE\_\_: TIM handle
- \_\_INTERRUPT\_\_: specifies the interrupt pending bit to clear.

**Return value:**

- None

`__HAL_TIM_IS_TIM_COUNTING_DOWN`

**Description:**

- Indicates whether or not the TIM Counter is used as downcounter.

**Parameters:**

- \_\_HANDLE\_\_: TIM handle.

**Return value:**

- False: (Counter used as upcounter) or True (Counter used as downcounter)

**Notes:**

- This macro is particularly usefull to get the counting mode when the timer operates in Center-aligned mode or Encoder mode.

`__HAL_TIM_SET_PRESCALER`

**Description:**

- Sets the TIM active prescaler register value on update event.

**Parameters:**

- \_\_HANDLE\_\_: TIM handle.
- \_\_PRESC\_\_: specifies the active prescaler register new value.

\_\_HAL\_TIM\_SET\_COUNTER**Return value:**

- None

**Description:**

- Sets the TIM Counter Register value on runtime.

**Parameters:**

- \_\_HANDLE\_\_: TIM handle.
- \_\_COUNTER\_\_: specifies the Counter register new value.

**Return value:**

- None

\_\_HAL\_TIM\_GET\_COUNTER**Description:**

- Gets the TIM Counter Register value on runtime.

**Parameters:**

- \_\_HANDLE\_\_: TIM handle.

**Return value:**

- 16-bit: or 32-bit value of the timer counter register (TIMx\_CNT)

\_\_HAL\_TIM\_SET\_AUTORELOAD**Description:**

- Sets the TIM Autoreload Register value on runtime without calling another time any Init function.

**Parameters:**

- \_\_HANDLE\_\_: TIM handle.
- \_\_AUTORELOAD\_\_: specifies the Counter register new value.

**Return value:**

- None

\_\_HAL\_TIM\_GET\_AUTORELOAD**Description:**

- Gets the TIM Autoreload Register value on runtime.

**Parameters:**

- \_\_HANDLE\_\_: TIM handle.

**Return value:**

- 16-bit: or 32-bit value of the timer auto-reload register(TIMx\_ARR)

[\\_\\_HAL\\_TIM\\_SET\\_CLOCKDIVISION](#)**Description:**

- Sets the TIM Clock Division value on runtime without calling another time any Init function.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): TIM handle.
- [\\_\\_CKD\\_\\_](#): specifies the clock division value. This parameter can be one of the following value:
  - [TIM\\_CLOCKDIVISION\\_DIV1](#): tDTS=tCK\_INT
  - [TIM\\_CLOCKDIVISION\\_DIV2](#): tDTS=2\*tCK\_INT
  - [TIM\\_CLOCKDIVISION\\_DIV4](#): tDTS=4\*tCK\_INT

**Return value:**

- None

[\\_\\_HAL\\_TIM\\_GET\\_CLOCKDIVISION](#)**Description:**

- Gets the TIM Clock Division value on runtime.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): TIM handle.

**Return value:**

- The: clock division can be one of the following values:
  - [TIM\\_CLOCKDIVISION\\_DIV1](#): tDTS=tCK\_INT
  - [TIM\\_CLOCKDIVISION\\_DIV2](#): tDTS=2\*tCK\_INT
  - [TIM\\_CLOCKDIVISION\\_DIV4](#): tDTS=4\*tCK\_INT

[\\_\\_HAL\\_TIM\\_SET\\_ICPRESCALER](#)**Description:**

- Sets the TIM Input Capture prescaler on runtime without calling another time

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): TIM handle.
- [\\_\\_CHANNEL\\_\\_](#): TIM Channels to be configured. This parameter can be one of the following values:
  - [TIM\\_CHANNEL\\_1](#): TIM Channel 1 selected
  - [TIM\\_CHANNEL\\_2](#): TIM Channel 2 selected
  - [TIM\\_CHANNEL\\_3](#): TIM Channel 3 selected
  - [TIM\\_CHANNEL\\_4](#): TIM Channel 4

- selected
  - ICPSC: specifies the Input Capture4 prescaler new value. This parameter can be one of the following values:
    - TIM\_ICPSC\_DIV1: no prescaler
    - TIM\_ICPSC\_DIV2: capture is done once every 2 events
    - TIM\_ICPSC\_DIV4: capture is done once every 4 events
    - TIM\_ICPSC\_DIV8: capture is done once every 8 events

**Return value:**

- None

**Description:**

- Gets the TIM Input Capture prescaler on runtime.

**Parameters:**

- HANDLE: TIM handle.
- CHANNEL: TIM Channels to be configured. This parameter can be one of the following values:
  - TIM\_CHANNEL\_1: get input capture 1 prescaler value
  - TIM\_CHANNEL\_2: get input capture 2 prescaler value
  - TIM\_CHANNEL\_3: get input capture 3 prescaler value
  - TIM\_CHANNEL\_4: get input capture 4 prescaler value

**Return value:**

- The input capture prescaler can be one of the following values:
  - TIM\_ICPSC\_DIV1: no prescaler
  - TIM\_ICPSC\_DIV2: capture is done once every 2 events
  - TIM\_ICPSC\_DIV4: capture is done once every 4 events
  - TIM\_ICPSC\_DIV8: capture is done once every 8 events

\_HAL\_TIM\_URS\_ENABLE

**Description:**

- Set the Update Request Source (URS) bit of the TIMx\_CR1 register.

**Parameters:**

- HANDLE: TIM handle.

**Return value:**

- None

**Notes:**

- When the USR bit of the TIMx\_CR1 register is set, only counter overflow/underflow generates an update interrupt or DMA request (if enabled)

[\\_\\_HAL\\_TIM\\_URS\\_DISABLE](#)**Description:**

- Reset the Update Request Source (URS) bit of the TIMx\_CR1 register.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): TIM handle.

**Return value:**

- None

**Notes:**

- When the USR bit of the TIMx\_CR1 register is reset, any of the following events generate an update interrupt or DMA request (if enabled): (+) Counter overflow/underflow (+) Setting the UG bit (+) Update generation through the slave mode controller

[\\_\\_HAL\\_TIM\\_SET\\_CAPTURE\\_POLARITY](#)**Description:**

- Sets the TIM Capture x input polarity on runtime.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): TIM handle.
- [\\_\\_CHANNEL\\_\\_](#): TIM Channels to be configured. This parameter can be one of the following values:
  - [TIM\\_CHANNEL\\_1](#): TIM Channel 1 selected
  - [TIM\\_CHANNEL\\_2](#): TIM Channel 2 selected
  - [TIM\\_CHANNEL\\_3](#): TIM Channel 3 selected
  - [TIM\\_CHANNEL\\_4](#): TIM Channel 4 selected
- [\\_\\_POLARITY\\_\\_](#): Polarity for TIx source
  - [TIM\\_INPUTCHANNELPOLARITY\\_RISING](#): Rising Edge
  - [TIM\\_INPUTCHANNELPOLARITY\\_FALLING](#): Falling Edge
  - [TIM\\_INPUTCHANNELPOLARITY\\_BOTHEDGE](#): Rising and Falling Edge

**Return value:**

- None

***TIM Flag Definition***

TIM\_FLAG\_UPDATE  
TIM\_FLAG\_CC1  
TIM\_FLAG\_CC2  
TIM\_FLAG\_CC3  
TIM\_FLAG\_CC4  
TIM\_FLAG\_COM  
TIM\_FLAG\_TRIGGER  
TIM\_FLAG\_BREAK  
TIM\_FLAG\_BREAK2  
TIM\_FLAG\_CC1OF  
TIM\_FLAG\_CC2OF  
TIM\_FLAG\_CC3OF  
TIM\_FLAG\_CC4OF

***TIM Input Capture Polarity***

TIM\_ICPOLARITY\_RISING  
TIM\_ICPOLARITY\_FALLING  
TIM\_ICPOLARITY\_BOTHEDGE

***TIM Input Capture Prescaler***

|                |                                                                      |
|----------------|----------------------------------------------------------------------|
| TIM_ICPSC_DIV1 | Capture performed each time an edge is detected on the capture input |
| TIM_ICPSC_DIV2 | Capture performed once every 2 events                                |
| TIM_ICPSC_DIV4 | Capture performed once every 4 events                                |
| TIM_ICPSC_DIV8 | Capture performed once every 8 events                                |

***TIM Input Capture Selection***

|                            |                                                                                            |
|----------------------------|--------------------------------------------------------------------------------------------|
| TIM_ICSELECTION_DIRECTTI   | TIM Input 1U, 2U, 3 or 4 is selected to be connected to IC1, IC2, IC3 or IC4, respectively |
| TIM_ICSELECTION_INDIRECTTI | TIM Input 1U, 2U, 3 or 4 is selected to be connected to IC2, IC1, IC4 or IC3, respectively |
| TIM_ICSELECTION_TRC        | TIM Input 1U, 2U, 3 or 4 is selected to be connected to TRC                                |

***TIM Input Channel Polarity***

|                                   |                         |
|-----------------------------------|-------------------------|
| TIM_INPUTCHANNELPOLARITY_RISING   | Polarity for TIx source |
| TIM_INPUTCHANNELPOLARITY_FALLING  | Polarity for TIx source |
| TIM_INPUTCHANNELPOLARITY_BOTHEDGE | Polarity for TIx source |

***TIM Interrupt Definition***

TIM\_IT\_UPDATE  
TIM\_IT\_CC1

TIM\_IT\_CC2  
TIM\_IT\_CC3  
TIM\_IT\_CC4  
TIM\_IT\_COM  
TIM\_IT\_TRIGGER  
TIM\_IT\_BREAK

***TIM Lock level***

TIM\_LOCKLEVEL\_OFF  
TIM\_LOCKLEVEL\_1  
TIM\_LOCKLEVEL\_2  
TIM\_LOCKLEVEL\_3

***TIM Master Mode Selection***

TIM\_TRGO\_RESET  
TIM\_TRGO\_ENABLE  
TIM\_TRGO\_UPDATE  
TIM\_TRGO\_OC1  
TIM\_TRGO\_OC1REF  
TIM\_TRGO\_OC2REF  
TIM\_TRGO\_OC3REF  
TIM\_TRGO\_OC4REF

***TIM Master Slave Mode***

TIM\_MASTERSLAVEMODE\_ENABLE  
TIM\_MASTERSLAVEMODE\_DISABLE

***TIM One Pulse Mode***

TIM\_OPMODE\_SINGLE  
TIM\_OPMODE\_REPETITIVE

***TIM OSSI Off State Selection for Idle mode state***

TIM\_OSSI\_ENABLE  
TIM\_OSSI\_DISABLE

***TIM OSSR Off State Selection for Run mode state***

TIM\_OSSR\_ENABLE  
TIM\_OSSR\_DISABLE

***TIM Output Compare Idle State***

TIM\_OCIDLESTATE\_SET  
TIM\_OCIDLESTATE\_RESET

***TIM Complementary Output Compare Idle State***

TIM\_OCNIDLESTATE\_SET

TIM\_OCNIDLESTATE\_RESET

***TIM Complementary Output Compare Polarity***

TIM\_OCPOLARITY\_HIGH

TIM\_OCPOLARITY\_LOW

***TIM Output Compare Polarity***

TIM\_OCPOLARITY\_HIGH

TIM\_OCPOLARITY\_LOW

***TIM Output Fast State***

TIM\_OCFAST\_DISABLE

TIM\_OCFAST\_ENABLE

***TIM TI1 Input Selection***

TIM\_TI1SELECTION\_CH1

TIM\_TI1SELECTION\_XORCOMBINATION

***TIM Trigger Polarity***

|                                 |                                               |
|---------------------------------|-----------------------------------------------|
| TIM_TRIGGERPOLARITY_INVERTED    | Polarity for ETRx trigger sources             |
| TIM_TRIGGERPOLARITY_NONINVERTED | Polarity for ETRx trigger sources             |
| TIM_TRIGGERPOLARITY_RISING      | Polarity for TIxFPx or TI1_ED trigger sources |
| TIM_TRIGGERPOLARITY_FALLING     | Polarity for TIxFPx or TI1_ED trigger sources |
| TIM_TRIGGERPOLARITY_BOTHEDGE    | Polarity for TIxFPx or TI1_ED trigger sources |

***TIM Trigger Prescaler***

TIM\_TRIGGERPRESCALER\_DIV1 No prescaler is used

TIM\_TRIGGERPRESCALER\_DIV2 Prescaler for External ETR Trigger: Capture performed once every 2 events.

TIM\_TRIGGERPRESCALER\_DIV4 Prescaler for External ETR Trigger: Capture performed once every 4 events.

TIM\_TRIGGERPRESCALER\_DIV8 Prescaler for External ETR Trigger: Capture performed once every 8 events.

***TIM Trigger Selection***

[TIM\\_TS\\_ITR0](#)  
[TIM\\_TS\\_ITR1](#)  
[TIM\\_TS\\_ITR2](#)  
[TIM\\_TS\\_ITR3](#)  
[TIM\\_TS\\_TI1F\\_ED](#)  
[TIM\\_TS\\_TI1FP1](#)  
[TIM\\_TS\\_TI2FP2](#)  
[TIM\\_TS\\_ETRF](#)  
[TIM\\_TS\\_NONE](#)

## 52 HAL TIM Extension Driver

### 52.1 TIME Firmware driver registers structures

#### 52.1.1 TIM\_HallSensor\_InitTypeDef

##### Data Fields

- *uint32\_t IC1Polarity*
- *uint32\_t IC1Prescaler*
- *uint32\_t IC1Filter*
- *uint32\_t Commutation\_Delay*

##### Field Documentation

- ***uint32\_t TIM\_HallSensor\_InitTypeDef::IC1Polarity***  
Specifies the active edge of the input signal. This parameter can be a value of [\*TIM\\_Input\\_Capture\\_Polarity\*](#)
- ***uint32\_t TIM\_HallSensor\_InitTypeDef::IC1Prescaler***  
Specifies the Input Capture Prescaler. This parameter can be a value of [\*TIM\\_Input\\_Capture\\_Prescaler\*](#)
- ***uint32\_t TIM\_HallSensor\_InitTypeDef::IC1Filter***  
Specifies the input capture filter. This parameter can be a number between Min\_Data = 0x0 and Max\_Data = 0xFU
- ***uint32\_t TIM\_HallSensor\_InitTypeDef::Commutation\_Delay***  
Specifies the pulse value to be loaded into the Capture Compare Register. This parameter can be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFFU

#### 52.1.2 TIM\_BreakDeadTimeConfigTypeDef

##### Data Fields

- *uint32\_t OffStateRunMode*
- *uint32\_t OffStateIDLEMode*
- *uint32\_t LockLevel*
- *uint32\_t DeadTime*
- *uint32\_t BreakState*
- *uint32\_t BreakPolarity*
- *uint32\_t BreakFilter*
- *uint32\_t Break2State*
- *uint32\_t Break2Polarity*
- *uint32\_t Break2Filter*
- *uint32\_t AutomaticOutput*

##### Field Documentation

- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::OffStateRunMode***  
TIM off state in run mode This parameter can be a value of [\*TIM\\_OSSR\\_Off\\_State\\_Selection\\_for\\_Run\\_mode\\_state\*](#)
- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::OffStateIDLEMode***  
TIM off state in IDLE mode This parameter can be a value of [\*TIM\\_OSSI\\_Off\\_State\\_Selection\\_for\\_Idle\\_mode\\_state\*](#)
- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::LockLevel***  
TIM Lock level This parameter can be a value of [\*TIM\\_Lock\\_level\*](#)

- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::DeadTime***  
TIM dead Time This parameter can be a number between Min\_Data = 0x00 and Max\_Data = 0xFFU
- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::BreakState***  
TIM Break State This parameter can be a value of ***TIM\_Break\_Input\_enable\_disable***
- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::BreakPolarity***  
TIM Break input polarity This parameter can be a value of ***TIM\_Break\_Polarity***
- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::BreakFilter***  
Specifies the break input filter. This parameter can be a number between Min\_Data = 0x0 and Max\_Data = 0xFU
- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::Break2State***  
TIM Break2 State This parameter can be a value of ***TIMEx\_Break2\_Input\_enable\_disable***
- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::Break2Polarity***  
TIM Break2 input polarity This parameter can be a value of ***TIMEx\_Break2\_Polarity***
- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::Break2Filter***  
TIM break2 input filter. This parameter can be a number between Min\_Data = 0x0 and Max\_Data = 0xFU
- ***uint32\_t TIM\_BreakDeadTimeConfigTypeDef::AutomaticOutput***  
TIM Automatic Output Enable state This parameter can be a value of ***TIM\_AOE\_Bit\_Set\_Reset***

### 52.1.3 TIM\_MasterConfigTypeDef

#### Data Fields

- ***uint32\_t MasterOutputTrigger***
- ***uint32\_t MasterOutputTrigger2***
- ***uint32\_t MasterSlaveMode***

#### Field Documentation

- ***uint32\_t TIM\_MasterConfigTypeDef::MasterOutputTrigger***  
Trigger output (TRGO) selection This parameter can be a value of ***TIM\_Master\_Mode\_Selection***
- ***uint32\_t TIM\_MasterConfigTypeDef::MasterOutputTrigger2***  
Trigger output2 (TRGO2) selection This parameter can be a value of ***TIMEx\_Master\_Mode\_Selection\_2***
- ***uint32\_t TIM\_MasterConfigTypeDef::MasterSlaveMode***  
Master/slave mode selection This parameter can be a value of ***TIM\_Master\_Slave\_Mode***

## 52.2 TIMEx Firmware driver API description

### 52.2.1 TIMER Extended features

The Timer Extended features include:

1. Complementary outputs with programmable dead-time for :
  - Output Compare
  - PWM generation (Edge and Center-aligned Mode)
  - One-pulse mode output
2. Synchronization circuit to control the timer with external signals and to interconnect several timers together.
3. Break input to put the timer output signals in reset state or in a known state.
4. Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes

## 52.2.2 How to use this driver

1. Initialize the TIM low level resources by implementing the following functions depending from feature used :
  - Complementary Output Compare : `HAL_TIM_OC_MspInit()`
  - Complementary PWM generation : `HAL_TIM_PWM_MspInit()`
  - Complementary One-pulse mode output : `HAL_TIM_OnePulse_MspInit()`
  - Hall Sensor output : `HAL_TIM_HallSensor_MspInit()`
2. Initialize the TIM low level resources :
  - a. Enable the TIM interface clock using `__HAL_RCC_TIMx_CLK_ENABLE()`;
  - b. TIM pins configuration
    - Enable the clock for the TIM GPIOs using the following function: `__HAL_RCC_GPIOx_CLK_ENABLE()`;
    - Configure these TIM pins in Alternate function mode using `HAL_GPIO_Init()`;
3. The external Clock can be configured, if needed (the default clock is the internal clock from the APBx), using the following function: `HAL_TIM_ConfigClockSource`, the clock configuration should be done before any start function.
4. Configure the TIM in the desired functioning mode using one of the initialization function of this driver:
  - `HAL_TIMEx_HallSensor_Init` and `HAL_TIMEx_ConfigCommutationEvent`: to use the Timer Hall Sensor Interface and the commutation event with the corresponding Interrupt and DMA request if needed (Note that One Timer is used to interface with the Hall sensor Interface and another Timer should be used to use the commutation event).
5. Activate the TIM peripheral using one of the start functions:
  - Complementary Output Compare : `HAL_TIMEx_OCN_Start()`, `HAL_TIMEx_OCN_Start_DMA()`, `HAL_TIMEx_OCN_Start_IT()`
  - Complementary PWM generation : `HAL_TIMEx_PWMN_Start()`, `HAL_TIMEx_PWMN_Start_DMA()`, `HAL_TIMEx_PWMN_Start_IT()`
  - Complementary One-pulse mode output : `HAL_TIMEx_OnePulseN_Start()`, `HAL_TIMEx_OnePulseN_Start_IT()`
  - Hall Sensor output : `HAL_TIMEx_HallSensor_Start()`, `HAL_TIMEx_HallSensor_Start_DMA()`, `HAL_TIMEx_HallSensor_Start_IT()`.

## 52.2.3 Timer Hall Sensor functions

This section provides functions allowing to:

- Initialize and configure TIM HAL Sensor.
- De-initialize TIM HAL Sensor.
- Start the Hall Sensor Interface.
- Stop the Hall Sensor Interface.
- Start the Hall Sensor Interface and enable interrupts.
- Stop the Hall Sensor Interface and disable interrupts.
- Start the Hall Sensor Interface and enable DMA transfers.
- Stop the Hall Sensor Interface and disable DMA transfers.

This section contains the following APIs:

- `HAL\_TIMEx\_HallSensor\_Init\(\)`
- `HAL\_TIMEx\_HallSensor\_DeInit\(\)`
- `HAL\_TIMEx\_HallSensor\_MspInit\(\)`
- `HAL\_TIMEx\_HallSensor\_MspDeInit\(\)`
- `HAL\_TIMEx\_HallSensor\_Start\(\)`
- `HAL\_TIMEx\_HallSensor\_Stop\(\)`
- `HAL\_TIMEx\_HallSensor\_Start\_IT\(\)`

- [\*HAL\\_TIMEx\\_HallSensor\\_Stop\\_IT\(\)\*](#)
- [\*HAL\\_TIMEx\\_HallSensor\\_Start\\_DMA\(\)\*](#)
- [\*HAL\\_TIMEx\\_HallSensor\\_Stop\\_DMA\(\)\*](#)

#### 52.2.4 Timer Complementary Output Compare functions

This section provides functions allowing to:

- Start the Complementary Output Compare.
- Stop the Complementary Output Compare.
- Start the Complementary Output Compare and enable interrupts.
- Stop the Complementary Output Compare and disable interrupts.
- Start the Complementary Output Compare and enable DMA transfers.
- Stop the Complementary Output Compare and disable DMA transfers.

This section contains the following APIs:

- [\*HAL\\_TIMEx\\_OCN\\_Start\(\)\*](#)
- [\*HAL\\_TIMEx\\_OCN\\_Stop\(\)\*](#)
- [\*HAL\\_TIMEx\\_OCN\\_Start\\_IT\(\)\*](#)
- [\*HAL\\_TIMEx\\_OCN\\_Stop\\_IT\(\)\*](#)
- [\*HAL\\_TIMEx\\_OCN\\_Start\\_DMA\(\)\*](#)
- [\*HAL\\_TIMEx\\_OCN\\_Stop\\_DMA\(\)\*](#)

#### 52.2.5 Timer Complementary PWM functions

This section provides functions allowing to:

- Start the Complementary PWM.
- Stop the Complementary PWM.
- Start the Complementary PWM and enable interrupts.
- Stop the Complementary PWM and disable interrupts.
- Start the Complementary PWM and enable DMA transfers.
- Stop the Complementary PWM and disable DMA transfers.
- Start the Complementary Input Capture measurement.
- Stop the Complementary Input Capture.
- Start the Complementary Input Capture and enable interrupts.
- Stop the Complementary Input Capture and disable interrupts.
- Start the Complementary Input Capture and enable DMA transfers.
- Stop the Complementary Input Capture and disable DMA transfers.
- Start the Complementary One Pulse generation.
- Stop the Complementary One Pulse.
- Start the Complementary One Pulse and enable interrupts.
- Stop the Complementary One Pulse and disable interrupts.

This section contains the following APIs:

- [\*HAL\\_TIMEx\\_PWMN\\_Start\(\)\*](#)
- [\*HAL\\_TIMEx\\_PWMN\\_Stop\(\)\*](#)
- [\*HAL\\_TIMEx\\_PWMN\\_Start\\_IT\(\)\*](#)
- [\*HAL\\_TIMEx\\_PWMN\\_Stop\\_IT\(\)\*](#)
- [\*HAL\\_TIMEx\\_PWMN\\_Start\\_DMA\(\)\*](#)
- [\*HAL\\_TIMEx\\_PWMN\\_Stop\\_DMA\(\)\*](#)

### 52.2.6 Timer Complementary One Pulse functions

This section provides functions allowing to:

- Start the Complementary One Pulse generation.
- Stop the Complementary One Pulse.
- Start the Complementary One Pulse and enable interrupts.
- Stop the Complementary One Pulse and disable interrupts.

This section contains the following APIs:

- [`HAL\_TIMEx\_OnePulseN\_Start\(\)`](#)
- [`HAL\_TIMEx\_OnePulseN\_Stop\(\)`](#)
- [`HAL\_TIMEx\_OnePulseN\_Start\_IT\(\)`](#)
- [`HAL\_TIMEx\_OnePulseN\_Stop\_IT\(\)`](#)

### 52.2.7 Peripheral Control functions

This section provides functions allowing to:

- Configure the commutation event in case of use of the Hall sensor interface.
- Configure Output channels for OC and PWM mode.
- Configure Complementary channels, break features and dead time.
- Configure Master synchronization.
- Configure timer remapping capabilities.
- Enable or disable channel grouping

This section contains the following APIs:

- [`HAL\_TIMEx\_ConfigCommuteEvent\(\)`](#)
- [`HAL\_TIMEx\_ConfigCommuteEvent\_IT\(\)`](#)
- [`HAL\_TIMEx\_ConfigCommuteEvent\_DMA\(\)`](#)
- [`HAL\_TIM\_OC\_ConfigChannel\(\)`](#)
- [`HAL\_TIM\_PWM\_ConfigChannel\(\)`](#)
- [`HAL\_TIMEx\_MasterConfigSynchronization\(\)`](#)
- [`HAL\_TIMEx\_ConfigBreakDeadTime\(\)`](#)
- [`HAL\_TIMEx\_RemapConfig\(\)`](#)
- [`HAL\_TIMEx\_GroupChannel5\(\)`](#)

### 52.2.8 Extended Callbacks functions

This section provides Extended TIM callback functions:

- Timer Commutation callback
- Timer Break callback

This section contains the following APIs:

- [`HAL\_TIMEx\_CommulationCallback\(\)`](#)
- [`HAL\_TIMEx\_BreakCallback\(\)`](#)
- [`HAL\_TIMEx\_Break2Callback\(\)`](#)

### 52.2.9 Extended Peripheral State functions

This subsection permit to get in run-time the status of the peripheral and the data flow.

This section contains the following APIs:

- [`HAL\_TIMEx\_HallSensor\_GetState\(\)`](#)

## 52.2.10 Detailed description of functions

### **HAL\_TIMEx\_HallSensor\_Init**

|                      |                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_HallSensor_Init<br/>(TIM_HandleTypeDef * htim, TIM_HallSensor_InitTypeDef * sConfig)</b>                                          |
| Function description | Initializes the TIM Hall Sensor Interface and create the associated handle.                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Encoder Interface handle</li> <li>• <b>sConfig:</b> TIM Hall Sensor configuration structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                           |

### **HAL\_TIMEx\_HallSensor\_DelInit**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_HallSensor_DelInit<br/>(TIM_HandleTypeDef * htim)</b>    |
| Function description | DeInitializes the TIM Hall Sensor interface.                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Hall Sensor handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                  |

### **HAL\_TIMEx\_HallSensor\_MspInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_TIMEx_HallSensor_MspInit (TIM_HandleTypeDef * htim)</b>         |
| Function description | Initializes the TIM Hall Sensor MSP.                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_TIMEx\_HallSensor\_MspDelInit**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_TIMEx_HallSensor_MspDelInit (TIM_HandleTypeDef * htim)</b>      |
| Function description | DeInitializes TIM Hall Sensor MSP.                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

### **HAL\_TIMEx\_HallSensor\_Start**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_HallSensor_Start<br/>(TIM_HandleTypeDef * htim)</b>      |
| Function description | Starts the TIM Hall Sensor Interface.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Hall Sensor handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                  |

**HAL\_TIMEx\_HallSensor\_Stop**

Function name      **HAL\_StatusTypeDef HAL\_TIMEx\_HallSensor\_Stop  
(TIM\_HandleTypeDef \* htim)**

Function description      Stops the TIM Hall sensor Interface.

Parameters      • **htim:** TIM Hall Sensor handle

Return values      • **HAL:** status

**HAL\_TIMEx\_HallSensor\_Start\_IT**

Function name      **HAL\_StatusTypeDef HAL\_TIMEx\_HallSensor\_Start\_IT  
(TIM\_HandleTypeDef \* htim)**

Function description      Starts the TIM Hall Sensor Interface in interrupt mode.

Parameters      • **htim:** TIM Hall Sensor handle

Return values      • **HAL:** status

**HAL\_TIMEx\_HallSensor\_Stop\_IT**

Function name      **HAL\_StatusTypeDef HAL\_TIMEx\_HallSensor\_Stop\_IT  
(TIM\_HandleTypeDef \* htim)**

Function description      Stops the TIM Hall Sensor Interface in interrupt mode.

Parameters      • **htim:** TIM handle

Return values      • **HAL:** status

**HAL\_TIMEx\_HallSensor\_Start\_DMA**

Function name      **HAL\_StatusTypeDef HAL\_TIMEx\_HallSensor\_Start\_DMA  
(TIM\_HandleTypeDef \* htim, uint32\_t \* pData, uint16\_t Length)**

Function description      Starts the TIM Hall Sensor Interface in DMA mode.

Parameters      • **htim:** TIM Hall Sensor handle

• **pData:** The destination Buffer address.

• **Length:** The length of data to be transferred from TIM peripheral to memory.

Return values      • **HAL:** status

**HAL\_TIMEx\_HallSensor\_Stop\_DMA**

Function name      **HAL\_StatusTypeDef HAL\_TIMEx\_HallSensor\_Stop\_DMA  
(TIM\_HandleTypeDef \* htim)**

Function description      Stops the TIM Hall Sensor Interface in DMA mode.

Parameters      • **htim:** TIM handle

Return values      • **HAL:** status

**HAL\_TIMEx\_OCN\_Start**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_OCN_Start<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                    |
| Function description | Starts the TIM Output Compare signal generation on the complementary output.                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Output Compare handle</li> <li>• <b>Channel:</b> TIM Channel to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                           |

**HAL\_TIMEx\_OCN\_Stop**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_OCN_Stop<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                       |
| Function description | Stops the TIM Output Compare signal generation on the complementary output.                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channel to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                             |

**HAL\_TIMEx\_OCN\_Start\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_OCN_Start_IT<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                     |
| Function description | Starts the TIM Output Compare signal generation in interrupt mode on the complementary output.                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM OC handle</li> <li>• <b>Channel:</b> TIM Channel to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                               |

**HAL\_TIMEx\_OCN\_Stop\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_OCN_Stop_IT<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Stops the TIM Output Compare signal generation in interrupt mode on the complementary output.                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Output Compare handle</li> <li>• <b>Channel:</b> TIM Channel to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                            |

**HAL\_TIMEx\_OCN\_Start\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_OCN_Start_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t Channel, uint32_t *<br/>pData, uint16_t Length)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Starts the TIM Output Compare signal generation in DMA mode on the complementary output.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Output Compare handle</li> <li>• <b>Channel:</b> TIM Channel to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> <li>• <b>pData:</b> The source Buffer address.</li> <li>• <b>Length:</b> The length of data to be transferred from memory to TIM peripheral</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**HAL\_TIMEx\_OCN\_Stop\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_OCN_Stop_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Stops the TIM Output Compare signal generation in DMA mode on the complementary output.                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Output Compare handle</li> <li>• <b>Channel:</b> TIM Channel to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                            |

**HAL\_TIMEx\_PWMN\_Start**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_PWMN_Start<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                    |
| Function description | Starts the PWM signal generation on the complementary output.                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channel to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                            |

**HAL\_TIMEx\_PWMN\_Stop**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_PWMN_Stop<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                      |
| Function description | Stops the PWM signal generation on the complementary output.                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channel to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                             |

**HAL\_TIMEx\_PWMN\_Start\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_PWMN_Start_IT<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                  |
| Function description | Starts the PWM signal generation in interrupt mode on the complementary output.                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channel to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                             |

**HAL\_TIMEx\_PWMN\_Stop\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_PWMN_Stop_IT<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                   |
| Function description | Stops the PWM signal generation in interrupt mode on the complementary output.                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channel to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                             |

**HAL\_TIMEx\_PWMN\_Start\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_PWMN_Start_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t Channel, uint32_t *<br/>pData, uint16_t Length)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Starts the TIM PWM signal generation in DMA mode on the complementary output.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channel to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> <li>• <b>pData:</b> The source Buffer address.</li> <li>• <b>Length:</b> The length of data to be transferred from memory to TIM peripheral</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

**HAL\_TIMEx\_PWMN\_Stop\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_PWMN_Stop_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t Channel)</b>                                                                                                                                                                                                                                                                                                                                  |
| Function description | Stops the TIM PWM signal generation in DMA mode on the complementary output.                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>Channel:</b> TIM Channel to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> <li>– TIM_CHANNEL_3: TIM Channel 3 selected</li> <li>– TIM_CHANNEL_4: TIM Channel 4 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                             |

**HAL\_TIMEx\_OnePulseN\_Start**

|                      |                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_OnePulseN_Start<br/>(TIM_HandleTypeDef * htim, uint32_t OutputChannel)</b>                                                                                                                                                                                                                                       |
| Function description | Starts the TIM One Pulse signal generation on the complementary output.                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM One Pulse handle</li> <li>• <b>OutputChannel:</b> TIM Channel to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                          |

**HAL\_TIMEx\_OnePulseN\_Stop**

|                      |                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_OnePulseN_Stop<br/>(TIM_HandleTypeDef * htim, uint32_t OutputChannel)</b>                                                                                                                                                                                                                                         |
| Function description | Stops the TIM One Pulse signal generation on the complementary output.                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM One Pulse handle</li> <li>• <b>OutputChannel:</b> TIM Channel to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                           |

**HAL\_TIMEx\_OnePulseN\_Start\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_OnePulseN_Start_IT<br/>(TIM_HandleTypeDef * htim, uint32_t OutputChannel)</b>                                                                                                                                                                                                                                    |
| Function description | Starts the TIM One Pulse signal generation in interrupt mode on the complementary channel.                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM One Pulse handle</li> <li>• <b>OutputChannel:</b> TIM Channel to be enabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> <li>– TIM_CHANNEL_2: TIM Channel 2 selected</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                          |

**HAL\_TIMEx\_OnePulseN\_Stop\_IT**

|                      |                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_OnePulseN_Stop_IT<br/>(TIM_HandleTypeDef * htim, uint32_t OutputChannel)</b>                                                                                                                                                                                     |
| Function description | Stops the TIM One Pulse signal generation in interrupt mode on the complementary channel.                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM One Pulse handle</li> <li>• <b>OutputChannel:</b> TIM Channel to be disabled This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– TIM_CHANNEL_1: TIM Channel 1 selected</li> </ul> </li> </ul> |

- TIM\_CHANNEL\_2: TIM Channel 2 selected
- **Return values**
- **HAL:** status

### **HAL\_TIMEx\_ConfigCommutationEvent**

|                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>        | <b>HAL_StatusTypeDef HAL_TIMEx_ConfigCommutationEvent<br/>(TIM_HandleTypeDef * htim, uint32_t InputTrigger, uint32_t CommutationSource)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Function description</b> | Configure the TIM commutation event sequence.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Parameters</b>           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>InputTrigger:</b> the Internal trigger corresponding to the Timer Interfacing with the Hall sensor This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_TS_ITR0: Internal trigger 0 selected</li> <li>- TIM_TS_ITR1: Internal trigger 1 selected</li> <li>- TIM_TS_ITR2: Internal trigger 2 selected</li> <li>- TIM_TS_ITR3: Internal trigger 3 selected</li> <li>- TIM_TS_NONE No trigger is needed</li> </ul> </li> <li>• <b>CommutationSource:</b> the Commutation Event source This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_COMMUTATION_TRGI: Commutation source is the TRGI of the Interface Timer</li> <li>- TIM_COMMUTATION_SOFTWARE: Commutation source is set by software using the COMG bit</li> </ul> </li> </ul> |
| <b>Return values</b>        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Notes</b>                | <ul style="list-style-type: none"> <li>• this function is mandatory to use the commutation event in order to update the configuration at each commutation detection on the TRGI input of the Timer, the typical use of this feature is with the use of another Timer(interface Timer) configured in Hall sensor interface, this interface Timer will generate the commutation at its TRGO output (connected to Timer used in this function) each time the TI1 of the Interface Timer detect a commutation at its input TI1.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                    |

### **HAL\_TIMEx\_ConfigCommutationEvent\_IT**

|                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>        | <b>HAL_StatusTypeDef<br/>HAL_TIMEx_ConfigCommutationEvent_IT<br/>(TIM_HandleTypeDef * htim, uint32_t InputTrigger, uint32_t CommutationSource)</b>                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Function description</b> | Configure the TIM commutation event sequence with interrupt.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Parameters</b>           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>InputTrigger:</b> the Internal trigger corresponding to the Timer Interfacing with the Hall sensor This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_TS_ITR0: Internal trigger 0 selected</li> <li>- TIM_TS_ITR1: Internal trigger 1 selected</li> <li>- TIM_TS_ITR2: Internal trigger 2 selected</li> <li>- TIM_TS_ITR3: Internal trigger 3 selected</li> <li>- TIM_TS_NONE No trigger is needed</li> </ul> </li> </ul> |

- **CommutationSource:** the Commutation Event source This parameter can be one of the following values:
  - TIM\_COMMUTATION\_TRGI: Commutation source is the TRGI of the Interface Timer
  - TIM\_COMMUTATION\_SOFTWARE: Commutation source is set by software using the COMG bit
- **HAL:** status
- this function is mandatory to use the commutation event in order to update the configuration at each commutation detection on the TRGI input of the Timer, the typical use of this feature is with the use of another Timer(interface Timer) configured in Hall sensor interface, this interface Timer will generate the commutation at its TRGO output (connected to Timer used in this function) each time the TI1 of the Interface Timer detect a commutation at its input TI1.

### **HAL\_TIMEx\_ConfigCommutationEvent\_DMA**

- |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef<br/>HAL_TIMEx_ConfigCommutationEvent_DMA<br/>(TIM_HandleTypeDef * htim, uint32_t InputTrigger, uint32_t CommutationSource)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Configure the TIM commutation event sequence with DMA.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>InputTrigger:</b> the Internal trigger corresponding to the Timer Interfacing with the Hall sensor This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– TIM_TS_ITR0: Internal trigger 0 selected</li> <li>– TIM_TS_ITR1: Internal trigger 1 selected</li> <li>– TIM_TS_ITR2: Internal trigger 2 selected</li> <li>– TIM_TS_ITR3: Internal trigger 3 selected</li> <li>– TIM_TS_NONE No trigger is needed</li> </ul> </li> <li>• <b>CommutationSource:</b> the Commutation Event source This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– TIM_COMMUTATION_TRGI: Commutation source is the TRGI of the Interface Timer</li> <li>– TIM_COMMUTATION_SOFTWARE: Commutation source is set by software using the COMG bit</li> </ul> </li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• this function is mandatory to use the commutation event in order to update the configuration at each commutation detection on the TRGI input of the Timer, the typical use of this feature is with the use of another Timer(interface Timer) configured in Hall sensor interface, this interface Timer will generate the commutation at its TRGO output (connected to Timer used in this function) each time the TI1 of the Interface Timer detect a commutation at its input TI1.</li> <li>• The user should configure the DMA in his own software, in This function only the COMDE bit is set</li> </ul>                                                                                                                                                                                                                                                                           |

**HAL\_TIMEx\_MasterConfigSynchronization**

|                      |                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_MasterConfigSynchronization(<br/>TIM_HandleTypeDef *htim, TIM_MasterConfigTypeDef *<br/>sMasterConfig)</b>                                                                                                |
| Function description | Configures the TIM in master mode.                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle.</li> <li>• <b>sMasterConfig:</b> pointer to a TIM_MasterConfigTypeDef structure that contains the selected trigger output (TRGO) and the Master/Slave mode.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                   |

**HAL\_TIMEx\_ConfigBreakDeadTime**

|                      |                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_ConfigBreakDeadTime(<br/>TIM_HandleTypeDef *htim,<br/>TIM_BreakDeadTimeConfigTypeDef *sBreakDeadTimeConfig)</b>                                                                                                                |
| Function description | Configures the Break feature, dead time, Lock level, OSSI/OSSR State and the AOE(automatic output enable).                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> <li>• <b>sBreakDeadTimeConfig:</b> pointer to a TIM_ConfigBreakDeadConfigTypeDef structure that contains the BDTR Register configuration information for the TIM peripheral.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• For STM32F302xC, STM32F302xE, STM32F303xC, STM32F358xx, STM32F303xE, STM32F398xx and STM32F303x8 two break inputs can be configured.</li> </ul>                                                                      |

**HAL\_TIMEx\_RemapConfig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_RemapConfig(<br/>TIM_HandleTypeDef *htim, uint32_t Remap1, uint32_t<br/>Remap2)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Configures the TIM1, TIM8 and TIM16 Remapping input capabilities.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle.</li> <li>• <b>Remap1:</b> specifies the first TIM remapping source. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- TIM_TIM1_ADC1_NONE TIM1_ETR is not connected to any AWD (analog watchdog)</li> <li>- TIM_TIM1_ADC1_AWD1: TIM1_ETR is connected to ADC1 AWD1</li> <li>- TIM_TIM1_ADC1_AWD2: TIM1_ETR is connected to ADC1 AWD2</li> <li>- TIM_TIM1_ADC1_AWD3: TIM1_ETR is connected to ADC1 AWD3</li> <li>- TIM_TIM8_ADC2_NONE TIM8_ETR is not connected to any AWD</li> </ul> </li> </ul> |

- TIM\_TIM8\_ADC2\_AWD1: TIM8\_ETR is connected to ADC2 AWD1
- TIM\_TIM8\_ADC2\_AWD2: TIM8\_ETR is connected to ADC2 AWD2
- TIM\_TIM8\_ADC2\_AWD3: TIM8\_ETR is connected to ADC2 AWD3
- TIM\_TIM16\_GPIO: TIM16 TI1 is connected to GPIO
- TIM\_TIM16\_RTC: TIM16 TI1 is connected to RTC clock
- TIM\_TIM16\_HSE: TIM16 TI1 is connected to HSE/32
- TIM\_TIM16\_MCO: TIM16 TI1 is connected to MCO
- **Remap2:** specifies the second TIMremapping source (if any). This parameter can be one of the following values:
  - TIM\_TIM1\_ADC4\_NONE TIM1\_ETR is not connected to any AWD (analog watchdog)
  - TIM\_TIM1\_ADC4\_AWD1: TIM1\_ETR is connected to ADC4 AWD1
  - TIM\_TIM1\_ADC4\_AWD2: TIM1\_ETR is connected to ADC4 AWD2
  - TIM\_TIM1\_ADC4\_AWD3: TIM1\_ETR is connected to ADC4 AWD3
  - TIM\_TIM8\_ADC3\_NONE TIM8\_ETR is not connected to any AWD
  - TIM\_TIM8\_ADC3\_AWD1: TIM8\_ETR is connected to ADC3 AWD1
  - TIM\_TIM8\_ADC3\_AWD2: TIM8\_ETR is connected to ADC3 AWD2
  - TIM\_TIM8\_ADC3\_AWD3: TIM8\_ETR is connected to ADC3 AWD3

**Return values**

- **HAL:** status

**HAL\_TIMEx\_GroupChannel5****Function name**

**HAL\_StatusTypeDef HAL\_TIMEx\_GroupChannel5  
(TIM\_HandleTypeDef \*htim, uint32\_t Channels)**

**Function description**

Group channel 5 and channel 1, 2 or 3.

**Parameters**

- **htim:** TIM handle.
- **Channels:** specifies the reference signal(s) the OC5REF is combined with. This parameter can be any combination of the following values: TIM\_GROUPCH5\_NONE No effect of OC5REF on OC1REFC, OC2REFC and OC3REFC  
TIM\_GROUPCH5\_OC1REFC: OC1REFC is the logical AND of OC1REFC and OC5REF  
TIM\_GROUPCH5\_OC2REFC: OC2REFC is the logical AND of OC2REFC and OC5REF  
TIM\_GROUPCH5\_OC3REFC: OC3REFC is the logical AND of OC3REFC and OC5REF

**Return values**

- **HAL:** status

**HAL\_TIMEx\_CommmutationCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_TIMEx_CommmutationCallback (TIM_HandleTypeDef * htim)</b>       |
| Function description | Hall commutation changed callback in non blocking mode.                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_TIMEx\_BreakCallback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_TIMEx_BreakCallback (TIM_HandleTypeDef * htim)</b>              |
| Function description | Hall Break detection callback in non blocking mode.                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_TIMEx\_Break2Callback**

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>void HAL_TIMEx_Break2Callback (TIM_HandleTypeDef * htim)</b>             |
| Function description | Hall Break2 detection callback in non blocking mode.                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>             |

**HAL\_TIMEx\_HallSensor\_GetState**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TIMEx_HallSensor_GetState (TIM_HandleTypeDef * htim)</b>       |
| Function description | Return the TIM Hall Sensor interface state.                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htim:</b> TIM Hall Sensor handle</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                   |

**TIMEx\_DMACommutationCplt**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b>void TIMEx_DMACommutationCplt (DMA_HandleTypeDef * hdma)</b>                         |
| Function description | TIM DMA Commutation callback.                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hdma:</b> pointer to DMA handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |

## 52.3 TIMEx Firmware driver defines

### 52.3.1 TIMEx

***TIMEX Break input 2 Enable***

TIM\_BREAK2\_DISABLE

TIM\_BREAK2\_ENABLE

***TIMEx Break Input 2 Polarity***

TIM\_BREAK2POLARITY\_LOW  
TIM\_BREAK2POLARITY\_HIGH

***TIMEx Channel***

TIM\_CHANNEL\_1  
TIM\_CHANNEL\_2  
TIM\_CHANNEL\_3  
TIM\_CHANNEL\_4  
TIM\_CHANNEL\_5  
TIM\_CHANNEL\_6  
TIM\_CHANNEL\_ALL

***TIMEx Clear Input Source***

TIM\_CLEARINPUTSOURCE\_ETR  
TIM\_CLEARINPUTSOURCE\_OCREFCLR  
TIM\_CLEARINPUTSOURCE\_NONE

***TIMEx Exported Macros***

\_HAL\_TIM\_SET\_COMPARE

**Description:**

- Sets the TIM Capture Compare Register value on runtime without calling another time ConfigChannel function.

**Parameters:**

- \_HANDLE\_: TIM handle.
- \_CHANNEL\_: TIM Channels to be configured. This parameter can be one of the following values:
  - TIM\_CHANNEL\_1: TIM Channel 1 selected
  - TIM\_CHANNEL\_2: TIM Channel 2 selected
  - TIM\_CHANNEL\_3: TIM Channel 3 selected
  - TIM\_CHANNEL\_4: TIM Channel 4 selected
  - TIM\_CHANNEL\_5: TIM Channel 5 selected
  - TIM\_CHANNEL\_6: TIM Channel 6 selected
- \_COMPARE\_: specifies the Capture Compare register new value.

**Return value:**

- None

\_HAL\_TIM\_GET\_COMPARE

**Description:**

- Gets the TIM Capture Compare Register

value on runtime.

**Parameters:**

- `__HANDLE__`: TIM handle.
- `__CHANNEL__`: TIM Channel associated with the capture compare register This parameter can be one of the following values:
  - `TIM_CHANNEL_1`: get capture/compare 1 register value
  - `TIM_CHANNEL_2`: get capture/compare 2 register value
  - `TIM_CHANNEL_3`: get capture/compare 3 register value
  - `TIM_CHANNEL_4`: get capture/compare 4 register value
  - `TIM_CHANNEL_5`: get capture/compare 5 register value
  - `TIM_CHANNEL_6`: get capture/compare 6 register value

**Return value:**

- None

`__HAL_TIM_ENABLE_OCxPRELOAD`

- Sets the TIM Output compare preload.

**Parameters:**

- `__HANDLE__`: TIM handle.
- `__CHANNEL__`: TIM Channels to be configured. This parameter can be one of the following values:
  - `TIM_CHANNEL_1`: TIM Channel 1 selected
  - `TIM_CHANNEL_2`: TIM Channel 2 selected
  - `TIM_CHANNEL_3`: TIM Channel 3 selected
  - `TIM_CHANNEL_4`: TIM Channel 4 selected
  - `TIM_CHANNEL_5`: TIM Channel 5 selected
  - `TIM_CHANNEL_6`: TIM Channel 6 selected

**Return value:**

- None

`__HAL_TIM_DISABLE_OCxPRELOAD`

- Resets the TIM Output compare preload.

**Parameters:**

- `__HANDLE__`: TIM handle.
- `__CHANNEL__`: TIM Channels to be

configured. This parameter can be one of the following values:

- TIM\_CHANNEL\_1: TIM Channel 1 selected
- TIM\_CHANNEL\_2: TIM Channel 2 selected
- TIM\_CHANNEL\_3: TIM Channel 3 selected
- TIM\_CHANNEL\_4: TIM Channel 4 selected
- TIM\_CHANNEL\_5: TIM Channel 5 selected
- TIM\_CHANNEL\_6: TIM Channel 6 selected

#### Return value:

- None

#### **Group Channel 5 and Channel 1U, 2 or 3**

|                      |                                                     |
|----------------------|-----------------------------------------------------|
| TIM_GROUPCH5_NONE    | No effect of OC5REF on OC1REFC, OC2REFC and OC3REFC |
| TIM_GROUPCH5_OC1REFC | OC1REFC is the logical AND of OC1REFC and OC5REF    |
| TIM_GROUPCH5_OC2REFC | OC2REFC is the logical AND of OC2REFC and OC5REF    |
| TIM_GROUPCH5_OC3REFC | OC3REFC is the logical AND of OC3REFC and OC5REF    |

#### **TIMEx Master Mode Selection 2 (TRGO2)**

|                                        |
|----------------------------------------|
| TIM_TRGO2_RESET                        |
| TIM_TRGO2_ENABLE                       |
| TIM_TRGO2_UPDATE                       |
| TIM_TRGO2_OC1                          |
| TIM_TRGO2_OC1REF                       |
| TIM_TRGO2_OC2REF                       |
| TIM_TRGO2_OC3REF                       |
| TIM_TRGO2_OC4REF                       |
| TIM_TRGO2_OC5REF                       |
| TIM_TRGO2_OC6REF                       |
| TIM_TRGO2_OC4REF_RISINGFALLING         |
| TIM_TRGO2_OC6REF_RISINGFALLING         |
| TIM_TRGO2_OC4REF_RISING_OC6REF_RISING  |
| TIM_TRGO2_OC4REF_RISING_OC6REF_FALLING |
| TIM_TRGO2_OC5REF_RISING_OC6REF_RISING  |
| TIM_TRGO2_OC5REF_RISING_OC6REF_FALLING |

#### **TIMEx Output Compare and PWM Modes**

|                   |
|-------------------|
| TIM_OCMODE_TIMING |
|-------------------|

TIM\_OCMODE\_ACTIVE  
TIM\_OCMODE\_INACTIVE  
TIM\_OCMODE\_TOGGLE  
TIM\_OCMODE\_PWM1  
TIM\_OCMODE\_PWM2  
TIM\_OCMODE\_FORCED\_ACTIVE  
TIM\_OCMODE\_FORCED\_INACTIVE  
TIM\_OCMODE\_RETRIGERRABLE\_OPM1  
TIM\_OCMODE\_RETRIGERRABLE\_OPM2  
TIM\_OCMODE\_COMBINED\_PWM1  
TIM\_OCMODE\_COMBINED\_PWM2  
TIM\_OCMODE\_ASSYMETRIC\_PWM1  
TIM\_OCMODE\_ASSYMETRIC\_PWM2

#### ***TIMEx Remapping 1***

|                    |                                                        |
|--------------------|--------------------------------------------------------|
| TIM_TIM1_ADC1_NONE | TIM1_ETR is not connected to any AWD (analog watchdog) |
| TIM_TIM1_ADC1_AWD1 | TIM1_ETR is connected to ADC1 AWD1                     |
| TIM_TIM1_ADC1_AWD2 | TIM1_ETR is connected to ADC1 AWD2                     |
| TIM_TIM1_ADC1_AWD3 | TIM1_ETR is connected to ADC1 AWD3                     |
| TIM_TIM8_ADC2_NONE | TIM8_ETR is not connected to any AWD (analog watchdog) |
| TIM_TIM8_ADC2_AWD1 | TIM8_ETR is connected to ADC2 AWD1                     |
| TIM_TIM8_ADC2_AWD2 | TIM8_ETR is connected to ADC2 AWD2                     |
| TIM_TIM8_ADC2_AWD3 | TIM8_ETR is connected to ADC2 AWD3                     |
| TIM_TIM16_GPIO     | TIM16 TI1 is connected to GPIO                         |
| TIM_TIM16_RTC      | TIM16 TI1 is connected to RTC_clock                    |
| TIM_TIM16_HSE      | TIM16 TI1 is connected to HSE/32U                      |
| TIM_TIM16_MCO      | TIM16 TI1 is connected to MCO                          |

#### ***TIMEx Remapping 2***

|                    |                                                        |
|--------------------|--------------------------------------------------------|
| TIM_TIM1_ADC4_NONE | TIM1_ETR is not connected to any AWD (analog watchdog) |
| TIM_TIM1_ADC4_AWD1 | TIM1_ETR is connected to ADC4 AWD1                     |
| TIM_TIM1_ADC4_AWD2 | TIM1_ETR is connected to ADC4 AWD2                     |
| TIM_TIM1_ADC4_AWD3 | TIM1_ETR is connected to ADC4 AWD3                     |
| TIM_TIM8_ADC3_NONE | TIM8_ETR is not connected to any AWD (analog watchdog) |
| TIM_TIM8_ADC3_AWD1 | TIM8_ETR is connected to ADC3 AWD1                     |
| TIM_TIM8_ADC3_AWD2 | TIM8_ETR is connected to ADC3 AWD2                     |
| TIM_TIM8_ADC3_AWD3 | TIM8_ETR is connected to ADC3 AWD3                     |
| TIM_TIM16_NONE     | Non significant value for TIM16U                       |

***TIMEx Slave mode***

TIM\_SLAVERESET\_DISABLE  
TIM\_SLAVERESET\_RESET  
TIM\_SLAVERESET\_GATED  
TIM\_SLAVERESET\_TRIGGER  
TIM\_SLAVERESET\_EXTERNAL1  
TIM\_SLAVERESET\_COMBINED\_RESETTRIGGER

## 53 HAL TSC Generic Driver

### 53.1 TSC Firmware driver registers structures

#### 53.1.1 TSC\_InitTypeDef

##### Data Fields

- *uint32\_t CTPulseHighLength*
- *uint32\_t CTPulseLowLength*
- *uint32\_t SpreadSpectrum*
- *uint32\_t SpreadSpectrumDeviation*
- *uint32\_t SpreadSpectrumPrescaler*
- *uint32\_t PulseGeneratorPrescaler*
- *uint32\_t MaxCountValue*
- *uint32\_t IODefaultMode*
- *uint32\_t SynchroPinPolarity*
- *uint32\_t AcquisitionMode*
- *uint32\_t MaxCountInterrupt*
- *uint32\_t ChannelIOs*
- *uint32\_t ShieldIOs*
- *uint32\_t SamplingIOs*

##### Field Documentation

- ***uint32\_t TSC\_InitTypeDef::CTPulseHighLength***  
Charge-transfer high pulse length This parameter can be a value of  
***TSC\_CTPulseHL\_Config***
- ***uint32\_t TSC\_InitTypeDef::CTPulseLowLength***  
Charge-transfer low pulse length This parameter can be a value of  
***TSC\_CTPulseLL\_Config***
- ***uint32\_t TSC\_InitTypeDef::SpreadSpectrum***  
Spread spectrum activation This parameter can be a value of  
***TSC\_CTPulseLL\_Config***
- ***uint32\_t TSC\_InitTypeDef::SpreadSpectrumDeviation***  
Spread spectrum deviation This parameter must be a number between Min\_Data = 0 and Max\_Data = 127U
- ***uint32\_t TSC\_InitTypeDef::SpreadSpectrumPrescaler***  
Spread spectrum prescaler This parameter can be a value of  
***TSC\_SpreadSpec\_Prescaler***
- ***uint32\_t TSC\_InitTypeDef::PulseGeneratorPrescaler***  
Pulse generator prescaler This parameter can be a value of  
***TSC\_PulseGenerator\_Prescaler***
- ***uint32\_t TSC\_InitTypeDef::MaxCountValue***  
Max count value This parameter can be a value of ***TSC\_MaxCount\_Value***
- ***uint32\_t TSC\_InitTypeDef::IODefaultMode***  
IO default mode This parameter can be a value of ***TSC\_IO\_Default\_Mode***
- ***uint32\_t TSC\_InitTypeDef::SynchroPinPolarity***  
Synchro pin polarity This parameter can be a value of ***TSC\_Synchro\_Pin\_Polarity***
- ***uint32\_t TSC\_InitTypeDef::AcquisitionMode***  
Acquisition mode This parameter can be a value of ***TSC\_Acquisition\_Mode***
- ***uint32\_t TSC\_InitTypeDef::MaxCountInterrupt***  
Max count interrupt activation This parameter can be set to ENABLE or DISABLE.

- *uint32\_t TSC\_InitTypeDef::ChannelIOs*  
Channel IOs mask
- *uint32\_t TSC\_InitTypeDef::ShieldIOs*  
Shield IOs mask
- *uint32\_t TSC\_InitTypeDef::SamplingIOs*  
Sampling IOs mask

### 53.1.2 TSC\_IOConfigTypeDef

#### Data Fields

- *uint32\_t ChannelIOs*
- *uint32\_t ShieldIOs*
- *uint32\_t SamplingIOs*

#### Field Documentation

- *uint32\_t TSC\_IOConfigTypeDef::ChannelIOs*  
Channel IOs mask
- *uint32\_t TSC\_IOConfigTypeDef::ShieldIOs*  
Shield IOs mask
- *uint32\_t TSC\_IOConfigTypeDef::SamplingIOs*  
Sampling IOs mask

### 53.1.3 TSC\_HandleTypeDef

#### Data Fields

- *TSC\_TypeDef \* Instance*
- *TSC\_InitTypeDef Init*
- *\_IO HAL\_TSC\_StateTypeDef State*
- *HAL\_LockTypeDef Lock*

#### Field Documentation

- *TSC\_TypeDef\* TSC\_HandleTypeDef::Instance*  
Register base address
- *TSC\_InitTypeDef TSC\_HandleTypeDef::Init*  
Initialization parameters
- *\_IO HAL\_TSC\_StateTypeDef TSC\_HandleTypeDef::State*  
Peripheral state
- *HAL\_LockTypeDef TSC\_HandleTypeDef::Lock*  
Lock feature

## 53.2 TSC Firmware driver API description

### 53.2.1 TSC specific features

1. Proven and robust surface charge transfer acquisition principle
2. Supports up to 3 capacitive sensing channels per group
3. Capacitive sensing channels can be acquired in parallel offering a very good response time
4. Spread spectrum feature to improve system robustness in noisy environments
5. Full hardware management of the charge transfer acquisition sequence
6. Programmable charge transfer frequency
7. Programmable sampling capacitor I/O pin
8. Programmable channel I/O pin
9. Programmable max count value to avoid long acquisition when a channel is faulty

10. Dedicated end of acquisition and max count error flags with interrupt capability
11. One sampling capacitor for up to 3 capacitive sensing channels to reduce the system components
12. Compatible with proximity, touchkey, linear and rotary touch sensor implementation

### 53.2.2 How to use this driver

1. Enable the TSC interface clock using `__HAL_RCC_TSC_CLK_ENABLE()` macro.
2. GPIO pins configuration
  - Enable the clock for the TSC GPIOs using `__HAL_RCC_GPIOx_CLK_ENABLE()` macro.
  - Configure the TSC pins used as sampling IOs in alternate function output Open-Drain mode, and TSC pins used as channel/shield IOs in alternate function output Push-Pull mode using `HAL_GPIO_Init()` function.
3. Interrupts configuration
  - Configure the NVIC (if the interrupt model is used) using `HAL_NVIC_SetPriority()` and `HAL_NVIC_EnableIRQ()` function.
4. TSC configuration
  - Configure all TSC parameters and used TSC IOs using `HAL_TSC_Init()` function.

#### Acquisition sequence

- Discharge all IOs using `HAL_TSC_IODischarge()` function.
- Wait a certain time allowing a good discharge of all capacitors. This delay depends of the sampling capacitor and electrodes design.
- Select the channel IOs to be acquired using `HAL_TSC_IOConfig()` function.
- Launch the acquisition using either `HAL_TSC_Start()` or `HAL_TSC_Start_IT()` function. If the synchronized mode is selected, the acquisition will start as soon as the signal is received on the synchro pin.
- Wait the end of acquisition using either `HAL_TSC_PollForAcquisition()` or `HAL_TSC_GetState()` function or using WFI instruction for example.
- Check the group acquisition status using `HAL_TSC_GroupGetStatus()` function.
- Read the acquisition value using `HAL_TSC_GroupGetValue()` function.

### 53.2.3 Initialization and de-initialization functions

This section provides functions allowing to:

- Initialize and configure the TSC.
- De-initialize the TSC.

This section contains the following APIs:

- [`HAL\_TSC\_Init\(\)`](#)
- [`HAL\_TSC\_DelInit\(\)`](#)
- [`HAL\_TSC\_MspInit\(\)`](#)
- [`HAL\_TSC\_MspDelInit\(\)`](#)

### 53.2.4 IO Operation functions

This section provides functions allowing to:

- Start acquisition in polling mode.
- Start acquisition in interrupt mode.
- Stop conversion in polling mode.
- Stop conversion in interrupt mode.
- Poll for acquisition completed.
- Get group acquisition status.

- Get group acquisition value.

This section contains the following APIs:

- [\*HAL\\_TSC\\_Start\(\)\*](#)
- [\*HAL\\_TSC\\_Start\\_IT\(\)\*](#)
- [\*HAL\\_TSC\\_Stop\(\)\*](#)
- [\*HAL\\_TSC\\_Stop\\_IT\(\)\*](#)
- [\*HAL\\_TSC\\_PollForAcquisition\(\)\*](#)
- [\*HAL\\_TSC\\_GroupGetStatus\(\)\*](#)
- [\*HAL\\_TSC\\_GroupGetValue\(\)\*](#)

### 53.2.5 Peripheral Control functions

This section provides functions allowing to:

- Configure TSC IOs
- Discharge TSC IOs

This section contains the following APIs:

- [\*HAL\\_TSC\\_IOConfig\(\)\*](#)
- [\*HAL\\_TSC\\_IODischarge\(\)\*](#)

### 53.2.6 State and Errors functions

This subsection provides functions allowing to

- Get TSC state.

This section contains the following APIs:

- [\*HAL\\_TSC\\_GetState\(\)\*](#)

### 53.2.7 Detailed description of functions

#### HAL\_TSC\_Init

|                      |                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TSC_Init (TSC_HandleTypeDef *htsc)</b>                                                                            |
| Function description | Initialize the TSC peripheral according to the specified parameters in the TSC_InitTypeDef structure and initialize the associated handle. |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htsc:</b> TSC handle</li></ul>                                                                  |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                       |

#### HAL\_TSC\_DeInit

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TSC_DeInit (TSC_HandleTypeDef *htsc)</b>         |
| Function description | Deinitialize the TSC peripheral registers to their default reset values.  |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htsc:</b> TSC handle</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>      |

**HAL\_TSC\_MspInit**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_TSC_MspInit (TSC_HandleTypeDef * htsc)</b>                                                                                                                     |
| Function description | Initialize the TSC MSP.                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                              |

**HAL\_TSC\_MspDeInit**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_TSC_MspDeInit (TSC_HandleTypeDef * htsc)</b>                                                                                                                   |
| Function description | Deinitialize the TSC MSP.                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                              |

**HAL\_TSC\_Start**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TSC_Start (TSC_HandleTypeDef * htsc)</b>                                                                                                          |
| Function description | Start the acquisition.                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                       |

**HAL\_TSC\_Start\_IT**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TSC_Start_IT (TSC_HandleTypeDef * htsc)</b>                                                                                                       |
| Function description | Start the acquisition in interrupt mode.                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status.</li></ul>                                                                                                      |

**HAL\_TSC\_Stop**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TSC_Stop (TSC_HandleTypeDef * htsc)</b>                                                                                                           |
| Function description | Stop the acquisition previously launched in polling mode.                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>                                                                                                       |

**HAL\_TSC\_Stop\_IT**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TSC_Stop_IT (TSC_HandleTypeDef * htsc)</b>                                                                                                          |
| Function description | Stop the acquisition previously launched in interrupt mode.                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                       |

**HAL\_TSC\_PollForAcquisition**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TSC_PollForAcquisition (TSC_HandleTypeDef * htsc)</b>                                                                                               |
| Function description | Start acquisition and wait until completion.                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• There is no need of a timeout parameter as the max count error is already managed by the TSC peripheral.</li> </ul>                 |

**HAL\_TSC\_GroupGetStatus**

|                      |                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>TSC_GroupStatusTypeDef HAL_TSC_GroupGetStatus (TSC_HandleTypeDef * htsc, uint32_t gx_index)</b>                                                                                                                          |
| Function description | Get the acquisition status for a group.                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li> <li>• <b>gx_index:</b> Index of the group</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Group:</b> status</li> </ul>                                                                                                                                                    |

**HAL\_TSC\_GroupGetValue**

|                      |                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_TSC_GroupGetValue (TSC_HandleTypeDef * htsc, uint32_t gx_index)</b>                                                                                                                                         |
| Function description | Get the acquisition measure for a group.                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li> <li>• <b>gx_index:</b> Index of the group</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Acquisition:</b> measure</li> </ul>                                                                                                                                             |

**HAL\_TSC\_IOConfig**

|                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TSC_IOConfig (TSC_HandleTypeDef * htsc, TSC_IOConfigTypeDef * config)</b>                                                                           |
| Function description | Configure TSC IOs.                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li> </ul> |

- **config:** pointer to the configuration structure.
- Return values      • **HAL:** status

### **HAL\_TSC\_IODischarge**

- |                      |                                                                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_TSC_IODischarge<br/>(TSC_HandleTypeDef * htsc, uint32_t choice)</b>                                                                                                                             |
| Function description | Discharge TSC IOs.                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li> <li>• <b>choice:</b> enable or disable</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                   |

### **HAL\_TSC\_GetState**

- |                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_TSC_StateTypeDef HAL_TSC_GetState<br/>(TSC_HandleTypeDef * htsc)</b>                                                                                                  |
| Function description | Return the TSC handle state.                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> state</li> </ul>                                                                                                        |

### **HAL\_TSC\_IRQHandler**

- |                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_TSC_IRQHandler (TSC_HandleTypeDef * htsc)</b>                                                                                                                    |
| Function description | Handle TSC interrupt request.                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

### **HAL\_TSC\_ConvCpltCallback**

- |                      |                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_TSC_ConvCpltCallback (TSC_HandleTypeDef * htsc)</b>                                                                                                              |
| Function description | Acquisition completed callback in non-blocking mode.                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**HAL\_TSC\_ErrorCallback**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_TSC_ErrorCallback (TSC_HandleTypeDef * htsc)</b>                                                                                                               |
| Function description | Error callback in non-blocking mode.                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"><li>• <b>htsc:</b> pointer to a TSC_HandleTypeDef structure that contains the configuration information for the specified TSC.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                              |

## 53.3 TSC Firmware driver defines

### 53.3.1 TSC

***Acquisition Mode***

TSC\_ACQ\_MODE\_NORMAL  
TSC\_ACQ\_MODE\_SYNCHRO

***CTPulse High Length***

TSC\_CTPH\_1CYCLE  
TSC\_CTPH\_2CYCLES  
TSC\_CTPH\_3CYCLES  
TSC\_CTPH\_4CYCLES  
TSC\_CTPH\_5CYCLES  
TSC\_CTPH\_6CYCLES  
TSC\_CTPH\_7CYCLES  
TSC\_CTPH\_8CYCLES  
TSC\_CTPH\_9CYCLES  
TSC\_CTPH\_10CYCLES  
TSC\_CTPH\_11CYCLES  
TSC\_CTPH\_12CYCLES  
TSC\_CTPH\_13CYCLES  
TSC\_CTPH\_14CYCLES  
TSC\_CTPH\_15CYCLES  
TSC\_CTPH\_16CYCLES

***CTPulse Low Length***

TSC\_CTPL\_1CYCLE  
TSC\_CTPL\_2CYCLES  
TSC\_CTPL\_3CYCLES  
TSC\_CTPL\_4CYCLES  
TSC\_CTPL\_5CYCLES  
TSC\_CTPL\_6CYCLES

TSC\_CTPL\_7CYCLES  
TSC\_CTPL\_8CYCLES  
TSC\_CTPL\_9CYCLES  
TSC\_CTPL\_10CYCLES  
TSC\_CTPL\_11CYCLES  
TSC\_CTPL\_12CYCLES  
TSC\_CTPL\_13CYCLES  
TSC\_CTPL\_14CYCLES  
TSC\_CTPL\_15CYCLES  
TSC\_CTPL\_16CYCLES

**TSC Exported Macros**

`__HAL_TSC_RESET_HANDLE_STATE`

**Description:**

- Reset TSC handle state.

**Parameters:**

- `__HANDLE__`: TSC handle

**Return value:**

- None

`__HAL_TSC_ENABLE`

**Description:**

- Enable the TSC peripheral.

**Parameters:**

- `__HANDLE__`: TSC handle

**Return value:**

- None

`__HAL_TSC_DISABLE`

**Description:**

- Disable the TSC peripheral.

**Parameters:**

- `__HANDLE__`: TSC handle

**Return value:**

- None

`__HAL_TSC_START_ACQ`

**Description:**

- Start acquisition.

**Parameters:**

- `__HANDLE__`: TSC handle

**Return value:**

- None

`__HAL_TSC_STOP_ACQ`**Description:**

- Stop acquisition.

**Parameters:**

- `__HANDLE__`: TSC handle

**Return value:**

- None

`__HAL_TSC_SET_IODEF_OUTPPLOW`**Description:**

- Set IO default mode to output push-pull low.

**Parameters:**

- `__HANDLE__`: TSC handle

**Return value:**

- None

`__HAL_TSC_SET_IODEF_INFLOAT`**Description:**

- Set IO default mode to input floating.

**Parameters:**

- `__HANDLE__`: TSC handle

**Return value:**

- None

`__HAL_TSC_SET_SYNC_POL_FALL`**Description:**

- Set synchronization polarity to falling edge.

**Parameters:**

- `__HANDLE__`: TSC handle

**Return value:**

- None

`__HAL_TSC_SET_SYNC_POL_RISE_HIGH`**Description:**

- Set synchronization polarity to rising edge and high level.

**Parameters:**

- `__HANDLE__`: TSC handle

**Return value:**

- None

`__HAL_TSC_ENABLE_IT`**Description:**

- Enable TSC interrupt.

**Parameters:**

- `__HANDLE__`: TSC handle

- `_INTERRUPT_`: TSC interrupt

**Description:**

- Disable TSC interrupt.

**Parameters:**

- `_HANDLE_`: TSC handle
- `_INTERRUPT_`: TSC interrupt

**Description:**

- Check whether the specified TSC interrupt source is enabled or not.

**Parameters:**

- `_HANDLE_`: TSC Handle
- `_INTERRUPT_`: TSC interrupt

**Description:**

- SET: or RESET

**Parameters:**

- `_HANDLE_`: TSC handle
- `_FLAG_`: TSC flag

**Description:**

- SET: or RESET

**Parameters:**

- `_HANDLE_`: TSC handle
- `_FLAG_`: TSC flag

**Description:**

- Clear the TSC's pending flag.

**Parameters:**

- `_HANDLE_`: TSC handle
- `_FLAG_`: TSC flag

**Description:**

- Enable schmitt trigger hysteresis on a group of IOs.

**Parameters:**

- `_HANDLE_`: TSC handle

- `__GX_IOY_MASK__`: IOs mask

**Description:**

- Disable schmitt trigger hysteresis on a group of IOs.

**Parameters:**

- `__HANDLE__`: TSC handle
- `__GX_IOY_MASK__`: IOs mask

**Return value:**

- None

**Description:**

- Open analog switch on a group of IOs.

**Parameters:**

- `__HANDLE__`: TSC handle
- `__GX_IOY_MASK__`: IOs mask

**Return value:**

- None

**Description:**

- Close analog switch on a group of IOs.

**Parameters:**

- `__HANDLE__`: TSC handle
- `__GX_IOY_MASK__`: IOs mask

**Return value:**

- None

**Description:**

- Enable a group of IOs in channel mode.

**Parameters:**

- `__HANDLE__`: TSC handle
- `__GX_IOY_MASK__`: IOs mask

**Return value:**

- None

**Description:**

- Disable a group of channel IOs.

**Parameters:**

- `__HANDLE__`: TSC handle
- `__GX_IOY_MASK__`: IOs mask

**Return value:**

- None

`__HAL_TSC_ENABLE_SAMPLING`

**Description:**

- Enable a group of IOs in sampling mode.

**Parameters:**

- `__HANDLE__`: TSC handle
- `__GX_IOY_MASK__`: IOs mask

**Return value:**

- None

`__HAL_TSC_DISABLE_SAMPLING`

**Description:**

- Disable a group of sampling IOs.

**Parameters:**

- `__HANDLE__`: TSC handle
- `__GX_IOY_MASK__`: IOs mask

**Return value:**

- None

`__HAL_TSC_ENABLE_GROUP`

**Description:**

- Enable acquisition groups.

**Parameters:**

- `__HANDLE__`: TSC handle
- `__GX_MASK__`: Groups mask

**Return value:**

- None

`__HAL_TSC_DISABLE_GROUP`

**Description:**

- Disable acquisition groups.

**Parameters:**

- `__HANDLE__`: TSC handle
- `__GX_MASK__`: Groups mask

**Return value:**

- None

`__HAL_TSC_GET_GROUP_STATUS`

**Description:**

- Gets acquisition group status.

**Parameters:**

- `__HANDLE__`: TSC Handle
- `__GX_INDEX__`: Group index

**Return value:**

- SET: or RESET

***Flags definition***

TSC\_FLAG\_EOA

TSC\_FLAG\_MCE

***Group definition***

TSC\_NB\_OF\_GROUPS

TSC\_GROUP1

TSC\_GROUP2

TSC\_GROUP3

TSC\_GROUP4

TSC\_GROUP5

TSC\_GROUP6

TSC\_GROUP7

TSC\_GROUP8

TSC\_ALL\_GROUPS

TSC\_GROUP1\_IDX

TSC\_GROUP2\_IDX

TSC\_GROUP3\_IDX

TSC\_GROUP4\_IDX

TSC\_GROUP5\_IDX

TSC\_GROUP6\_IDX

TSC\_GROUP7\_IDX

TSC\_GROUP8\_IDX

TSC\_GROUP1\_IO1

TSC\_GROUP1\_IO2

TSC\_GROUP1\_IO3

TSC\_GROUP1\_IO4

TSC\_GROUP1\_ALL\_IOS

TSC\_GROUP2\_IO1

TSC\_GROUP2\_IO2

TSC\_GROUP2\_IO3

TSC\_GROUP2\_IO4

TSC\_GROUP2\_ALL\_IOS

TSC\_GROUP3\_IO1

TSC\_GROUP3\_IO2

TSC\_GROUP3\_IO3

TSC\_GROUP3\_IO4

TSC\_GROUP3\_ALL\_IOS  
TSC\_GROUP4\_IO1  
TSC\_GROUP4\_IO2  
TSC\_GROUP4\_IO3  
TSC\_GROUP4\_IO4  
TSC\_GROUP4\_ALL\_IOS  
TSC\_GROUP5\_IO1  
TSC\_GROUP5\_IO2  
TSC\_GROUP5\_IO3  
TSC\_GROUP5\_IO4  
TSC\_GROUP5\_ALL\_IOS  
TSC\_GROUP6\_IO1  
TSC\_GROUP6\_IO2  
TSC\_GROUP6\_IO3  
TSC\_GROUP6\_IO4  
TSC\_GROUP6\_ALL\_IOS  
TSC\_GROUP7\_IO1  
TSC\_GROUP7\_IO2  
TSC\_GROUP7\_IO3  
TSC\_GROUP7\_IO4  
TSC\_GROUP7\_ALL\_IOS  
TSC\_GROUP8\_IO1  
TSC\_GROUP8\_IO2  
TSC\_GROUP8\_IO3  
TSC\_GROUP8\_IO4  
TSC\_GROUP8\_ALL\_IOS  
TSC\_ALL\_GROUPS\_ALL\_IOS

***Interrupts definition***

TSC\_IT\_EOA  
TSC\_IT\_MCE

***IO Default Mode***

TSC\_IODEF\_OUT\_PP\_LOW  
TSC\_IODEF\_IN\_FLOAT

***IO Mode***

TSC\_IOMODE\_UNUSED  
TSC\_IOMODE\_CHANNEL

TSC\_IOMODE\_SHIELD

TSC\_IOMODE\_SAMPLING

***Max Count Value***

TSC\_MCV\_255

TSC\_MCV\_511

TSC\_MCV\_1023

TSC\_MCV\_2047

TSC\_MCV\_4095

TSC\_MCV\_8191

TSC\_MCV\_16383

***Pulse Generator Prescaler***

TSC\_PG\_PRESC\_DIV1

TSC\_PG\_PRESC\_DIV2

TSC\_PG\_PRESC\_DIV4

TSC\_PG\_PRESC\_DIV8

TSC\_PG\_PRESC\_DIV16

TSC\_PG\_PRESC\_DIV32

TSC\_PG\_PRESC\_DIV64

TSC\_PG\_PRESC\_DIV128

***Spread Spectrum Prescaler***

TSC\_SS\_PRESC\_DIV1

TSC\_SS\_PRESC\_DIV2

***Synchro Pin Polarity***

TSC\_SYNC\_POLARITY\_FALLING

TSC\_SYNC\_POLARITY\_RISING

## 54 HAL UART Generic Driver

### 54.1 UART Firmware driver registers structures

#### 54.1.1 **UART\_InitTypeDef**

##### Data Fields

- *uint32\_t BaudRate*
- *uint32\_t WordLength*
- *uint32\_t StopBits*
- *uint32\_t Parity*
- *uint32\_t Mode*
- *uint32\_t HwFlowCtl*
- *uint32\_t OverSampling*
- *uint32\_t OneBitSampling*

UART, pour Universal Asynchronous Receiver Transmitter,

##### Field Documentation

- ***uint32\_t UART\_InitTypeDef::BaudRate***

This member configures the UART communication baud rate. The baud rate register is computed using the following formula:  
If oversampling is 16 or in LIN mode, Baud Rate Register = ((PCLKx) / ((huart->Init.BaudRate)))  
If oversampling is 8U, Baud Rate Register[15:4] = ((2U \* PCLKx) / ((huart->Init.BaudRate)))[15:4] Baud Rate Register[3] = 0  
Baud Rate Register[2:0] = (((2U \* PCLKx) / ((huart->Init.BaudRate)))[3:0]) >> 1

- ***uint32\_t UART\_InitTypeDef::WordLength***

Specifies the number of data bits transmitted or received in a frame. This parameter can be a value of [\*\*UARTEx\\_Word\\_Length\*\*](#).

- ***uint32\_t UART\_InitTypeDef::StopBits***

Specifies the number of stop bits transmitted. This parameter can be a value of [\*\*UART\\_Stop\\_Bits\*\*](#).

- ***uint32\_t UART\_InitTypeDef::Parity***

Specifies the parity mode. This parameter can be a value of [\*\*UART\\_Parity\*\*](#)  
**Note:**When parity is enabled, the computed parity is inserted at the MSB position of the transmitted data (9th bit when the word length is set to 9 data bits; 8th bit when the word length is set to 8 data bits).

- ***uint32\_t UART\_InitTypeDef::Mode***

Specifies whether the Receive or Transmit mode is enabled or disabled. This parameter can be a value of [\*\*UART\\_Mode\*\*](#).

- ***uint32\_t UART\_InitTypeDef::HwFlowCtl***

Specifies whether the hardware flow control mode is enabled or disabled. This parameter can be a value of [\*\*UART\\_Hardware\\_Flow\\_Control\*\*](#).

- ***uint32\_t UART\_InitTypeDef::OverSampling***

Specifies whether the Over sampling 8 is enabled or disabled, to achieve higher speed (up to f\_PCLK/8U). This parameter can be a value of [\*\*UART\\_Over\\_Sampling\*\*](#).

- ***uint32\_t UART\_InitTypeDef::OneBitSampling***

Specifies whether a single sample or three samples' majority vote is selected. Selecting the single sample method increases the receiver tolerance to clock deviations. This parameter can be a value of [\*\*UART\\_OneBit\\_Sampling\*\*](#).

#### 54.1.2 **UART\_AdvFeatureInitTypeDef**

##### Data Fields

- `uint32_t AdvFeatureInit`
- `uint32_t TxPinLevInvert`
- `uint32_t RxPinLevInvert`
- `uint32_t DataInvert`
- `uint32_t Swap`
- `uint32_t OverrunDisable`
- `uint32_t DMADisableonRxError`
- `uint32_t AutoBaudRateEnable`
- `uint32_t AutoBaudRateMode`
- `uint32_t MSBFirst`

#### Field Documentation

- **`uint32_t UART_AdvFeatureInitTypeDef::AdvFeatureInit`**  
Specifies which advanced UART features is initialized. Several Advanced Features may be initialized at the same time . This parameter can be a value of [`UART\_Advanced\_Features\_Initialization\_Type`](#).
- **`uint32_t UART_AdvFeatureInitTypeDef::TxPinLevInvert`**  
Specifies whether the TX pin active level is inverted. This parameter can be a value of [`UART\_Tx\_Inv`](#).
- **`uint32_t UART_AdvFeatureInitTypeDef::RxPinLevInvert`**  
Specifies whether the RX pin active level is inverted. This parameter can be a value of [`UART\_Rx\_Inv`](#).
- **`uint32_t UART_AdvFeatureInitTypeDef::DataInvert`**  
Specifies whether data are inverted (positive/direct logic vs negative/inverted logic). This parameter can be a value of [`UART\_Data\_Inv`](#).
- **`uint32_t UART_AdvFeatureInitTypeDef::Swap`**  
Specifies whether TX and RX pins are swapped. This parameter can be a value of [`UART\_Rx\_Tx\_Swap`](#).
- **`uint32_t UART_AdvFeatureInitTypeDef::OverrunDisable`**  
Specifies whether the reception overrun detection is disabled. This parameter can be a value of [`UART\_Overrun\_Disable`](#).
- **`uint32_t UART_AdvFeatureInitTypeDef::DMADisableonRxError`**  
Specifies whether the DMA is disabled in case of reception error. This parameter can be a value of [`UART\_DMA\_Disable\_on\_Rx\_Error`](#).
- **`uint32_t UART_AdvFeatureInitTypeDef::AutoBaudRateEnable`**  
Specifies whether auto Baud rate detection is enabled. This parameter can be a value of [`UART\_AutoBaudRate\_Enable`](#)
- **`uint32_t UART_AdvFeatureInitTypeDef::AutoBaudRateMode`**  
If auto Baud rate detection is enabled, specifies how the rate detection is carried out. This parameter can be a value of [`UART\_AutoBaud\_Rate\_Mode`](#).
- **`uint32_t UART_AdvFeatureInitTypeDef::MSBFirst`**  
Specifies whether MSB is sent first on UART line. This parameter can be a value of [`UART\_MSB\_First`](#).

### 54.1.3 `UART_WakeUpTypeDef`

#### Data Fields

- `uint32_t WakeUpEvent`
- `uint16_t AddressLength`
- `uint8_t Address`

#### Field Documentation

- **`uint32_t UART_WakeUpTypeDef::WakeUpEvent`**  
Specifies which event will activat the Wakeup from Stop mode flag (WUF). This

- parameter can be a value of [\*\*UART\\_WakeUp\\_from\\_Stop\\_Selection\*\*](#). If set to **UART\_WAKEUP\_ON\_ADDRESS**, the two other fields below must be filled up.
- ***uint16\_t* **UART\_WakeUpTypeDef::AddressLength****  
Specifies whether the address is 4 or 7-bit long. This parameter can be a value of [\*\*UART\\_WakeUp\\_Address\\_Length\*\*](#).
  - ***uint8\_t* **UART\_WakeUpTypeDef::Address****  
UART/USART node address (7-bit long max).

#### 54.1.4 **UART\_HandleTypeDef**

##### Data Fields

- ***USART\_TypeDef \** Instance**
- ***UART\_InitTypeDef Init***
- ***UART\_AdvFeatureInitTypeDef AdvancedInit***
- ***uint8\_t \* pTxBuffPtr***
- ***uint16\_t TxXferSize***
- ***\_IO uint16\_t TxXferCount***
- ***uint8\_t \* pRxBuffPtr***
- ***uint16\_t RxXferSize***
- ***\_IO uint16\_t RxXferCount***
- ***uint16\_t Mask***
- ***DMA\_HandleTypeDef \* hdmatx***
- ***DMA\_HandleTypeDef \* hdmarx***
- ***HAL\_LockTypeDef Lock***
- ***\_IO HAL\_UART\_StateTypeDef gState***
- ***\_IO HAL\_UART\_StateTypeDef RxState***
- ***\_IO uint32\_t ErrorCode***

##### Field Documentation

- ***USART\_TypeDef\* UART\_HandleTypeDef::Instance***  
UART registers base address
- ***UART\_InitTypeDef UART\_HandleTypeDef::Init***  
UART communication parameters
- ***UART\_AdvFeatureInitTypeDef UART\_HandleTypeDef::AdvancedInit***  
UART Advanced Features initialization parameters
- ***uint8\_t\* UART\_HandleTypeDef::pTxBuffPtr***  
Pointer to UART Tx transfer Buffer
- ***uint16\_t UART\_HandleTypeDef::TxXferSize***  
UART Tx Transfer size
- ***\_IO uint16\_t UART\_HandleTypeDef::TxXferCount***  
UART Tx Transfer Counter
- ***uint8\_t\* UART\_HandleTypeDef::pRxBuffPtr***  
Pointer to UART Rx transfer Buffer
- ***uint16\_t UART\_HandleTypeDef::RxXferSize***  
UART Rx Transfer size
- ***\_IO uint16\_t UART\_HandleTypeDef::RxXferCount***  
UART Rx Transfer Counter
- ***uint16\_t UART\_HandleTypeDef::Mask***  
UART Rx RDR register mask
- ***DMA\_HandleTypeDef\* UART\_HandleTypeDef::hdmatx***  
UART Tx DMA Handle parameters
- ***DMA\_HandleTypeDef\* UART\_HandleTypeDef::hdmarx***  
UART Rx DMA Handle parameters

- **`HAL_LockTypeDef` `UART_HandleTypeDef::Lock`**  
Locking object
- **`_IO HAL_UART_StateTypeDef` `UART_HandleTypeDef::gState`**  
UART state information related to global Handle management and also related to Tx operations. This parameter can be a value of `HAL_UART_StateTypeDef`
- **`_IO HAL_UART_StateTypeDef` `UART_HandleTypeDef::RxState`**  
UART state information related to Rx operations. This parameter can be a value of `HAL_UART_StateTypeDef`
- **`_IO uint32_t` `UART_HandleTypeDef::ErrorCode`**  
UART Error code

## 54.2 UART Firmware driver API description

### 54.2.1 How to use this driver

The UART HAL driver can be used as follows:

1. Declare a `UART_HandleTypeDef` handle structure (eg. `UART_HandleTypeDef huart`).
2. Initialize the UART low level resources by implementing the `HAL_UART_MspInit()` API:
  - Enable the USARTx interface clock.
  - UART pins configuration:
    - Enable the clock for the UART GPIOs.
    - Configure these UART pins as alternate function pull-up.
  - NVIC configuration if you need to use interrupt process (`HAL_UART_Transmit_IT()` and `HAL_UART_Receive_IT()` APIs):
    - Configure the USARTx interrupt priority.
    - Enable the NVIC USART IRQ handle.
  - UART interrupts handling: The specific UART interrupts (Transmission complete interrupt, RXNE interrupt and Error Interrupts) are managed using the macros `__HAL_UART_ENABLE_IT()` and `__HAL_UART_DISABLE_IT()` inside the transmit and receive processes.
  - DMA Configuration if you need to use DMA process (`HAL_UART_Transmit_DMA()` and `HAL_UART_Receive_DMA()` APIs):
    - Declare a DMA handle structure for the Tx/Rx channel.
    - Enable the DMAx interface clock.
    - Configure the declared DMA handle structure with the required Tx/Rx parameters.
    - Configure the DMA Tx/Rx channel.
    - Associate the initialized DMA handle to the UART DMA Tx/Rx handle.
    - Configure the priority and enable the NVIC for the transfer complete interrupt on the DMA Tx/Rx channel.
3. Program the Baud Rate, Word Length, Stop Bit, Parity, Hardware flow control and Mode (Receiver/Transmitter) in the `huart` handle Init structure.
4. If required, program UART advanced features (TX/RX pins swap, auto Baud rate detection,...) in the `huart` handle AdvancedInit structure.
5. For the UART asynchronous mode, initialize the UART registers by calling the `HAL_UART_Init()` API.
6. For the UART Half duplex mode, initialize the UART registers by calling the `HAL_HalfDuplex_Init()` API.
7. For the UART LIN (Local Interconnection Network) mode, initialize the UART registers by calling the `HAL_LIN_Init()` API.
8. For the UART Multiprocessor mode, initialize the UART registers by calling the `HAL_MultiProcessor_Init()` API.

9. For the UART RS485 Driver Enabled mode, initialize the UART registers by calling the HAL\_RS485Ex\_Init() API.



These APIs (HAL\_UART\_Init(), HAL\_HalfDuplex\_Init(), HAL\_MultiProcessor\_Init()), also configure the low level Hardware (GPIO, CLOCK, CORTEX...etc) by calling the customized HAL\_UART\_MspInit() API.

Three operation modes are available within this driver :

### Polling mode IO operation

- Send an amount of data in blocking mode using HAL\_UART\_Transmit()
- Receive an amount of data in blocking mode using HAL\_UART\_Receive()

### Interrupt mode IO operation

- Send an amount of data in non blocking mode using HAL\_UART\_Transmit\_IT()
- At transmission end of half transfer HAL\_UART\_TxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_UART\_TxHalfCpltCallback
- At transmission end of transfer HAL\_UART\_TxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_UART\_TxCpltCallback
- Receive an amount of data in non blocking mode using HAL\_UART\_Receive\_IT()
- At reception end of half transfer HAL\_UART\_RxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_UART\_RxHalfCpltCallback
- At reception end of transfer HAL\_UART\_RxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_UART\_RxCpltCallback
- In case of transfer Error, HAL\_UART\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_UART\_ErrorCallback

### DMA mode IO operation

- Send an amount of data in non blocking mode (DMA) using HAL\_UART\_Transmit\_DMA()
- At transmission end of half transfer HAL\_UART\_TxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_UART\_TxHalfCpltCallback
- At transmission end of transfer HAL\_UART\_TxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_UART\_TxCpltCallback
- Receive an amount of data in non blocking mode (DMA) using HAL\_UART\_Receive\_DMA()
- At reception end of half transfer HAL\_UART\_RxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_UART\_RxHalfCpltCallback
- At reception end of transfer HAL\_UART\_RxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_UART\_RxCpltCallback
- In case of transfer Error, HAL\_UART\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_UART\_ErrorCallback
- Pause the DMA Transfer using HAL\_UART\_DMAPause()
- Resume the DMA Transfer using HAL\_UART\_DMAResume()
- Stop the DMA Transfer using HAL\_UART\_DMAStop()

### UART HAL driver macros list

Below the list of most used macros in UART HAL driver.

- `_HAL_UART_ENABLE`: Enable the UART peripheral
- `_HAL_UART_DISABLE`: Disable the UART peripheral
- `_HAL_UART_GET_FLAG` : Check whether the specified UART flag is set or not
- `_HAL_UART_CLEAR_FLAG` : Clear the specified UART pending flag
- `_HAL_UART_ENABLE_IT`: Enable the specified UART interrupt
- `_HAL_UART_DISABLE_IT`: Disable the specified UART interrupt



You can refer to the UART HAL driver header file for more useful macros

### 54.2.2 Initialization and Configuration functions

This subsection provides a set of functions allowing to initialize the USARTx or the UARTy in asynchronous mode.

- For the asynchronous mode the parameters below can be configured:
  - Baud Rate
  - Word Length
  - Stop Bit
  - Parity
  - Hardware flow control
  - Receiver/transmitter modes
  - Over Sampling Method
  - One-Bit Sampling Method
- For the asynchronous mode, the following advanced features can be configured as well:
  - TX and/or RX pin level inversion
  - data logical level inversion
  - RX and TX pins swap
  - RX overrun detection disabling
  - DMA disabling on RX error
  - MSB first on communication line
  - auto Baud rate detection

The `HAL_UART_Init()`, `HAL_HalfDuplex_Init()`, `HAL_LIN_Init()` and `HAL_MultiProcessor_Init()` API follow respectively the UART asynchronous, UART Half duplex, UART LIN mode and multiprocessor configuration procedures (details for the procedures are available in reference manual).

This section contains the following APIs:

- [`HAL\_UART\_Init\(\)`](#)
- [`HAL\_HalfDuplex\_Init\(\)`](#)
- [`HAL\_LIN\_Init\(\)`](#)
- [`HAL\_MultiProcessor\_Init\(\)`](#)
- [`HAL\_UART\_DeInit\(\)`](#)
- [`HAL\_UART\_MspInit\(\)`](#)
- [`HAL\_UART\_MspDeInit\(\)`](#)

### 54.2.3 IO operation functions

This section contains the following APIs:

- [\*HAL\\_UART\\_Transmit\(\)\*](#)
- [\*HAL\\_UART\\_Receive\(\)\*](#)
- [\*HAL\\_UART\\_Transmit\\_IT\(\)\*](#)
- [\*HAL\\_UART\\_Receive\\_IT\(\)\*](#)
- [\*HAL\\_UART\\_Transmit\\_DMA\(\)\*](#)
- [\*HAL\\_UART\\_Receive\\_DMA\(\)\*](#)
- [\*HAL\\_UART\\_DMAPause\(\)\*](#)
- [\*HAL\\_UART\\_DMAResume\(\)\*](#)
- [\*HAL\\_UART\\_DMAStop\(\)\*](#)
- [\*HAL\\_UART\\_Abort\(\)\*](#)
- [\*HAL\\_UART\\_AbortTransmit\(\)\*](#)
- [\*HAL\\_UART\\_AbortReceive\(\)\*](#)
- [\*HAL\\_UART\\_Abort\\_IT\(\)\*](#)
- [\*HAL\\_UART\\_AbortTransmit\\_IT\(\)\*](#)
- [\*HAL\\_UART\\_AbortReceive\\_IT\(\)\*](#)
- [\*HAL\\_UART\\_IRQHandler\(\)\*](#)
- [\*HAL\\_UART\\_TxCpltCallback\(\)\*](#)
- [\*HAL\\_UART\\_TxHalfCpltCallback\(\)\*](#)
- [\*HAL\\_UART\\_RxCpltCallback\(\)\*](#)
- [\*HAL\\_UART\\_RxHalfCpltCallback\(\)\*](#)
- [\*HAL\\_UART\\_ErrorCallback\(\)\*](#)
- [\*HAL\\_UART\\_AbortCpltCallback\(\)\*](#)
- [\*HAL\\_UART\\_AbortTransmitCpltCallback\(\)\*](#)
- [\*HAL\\_UART\\_AbortReceiveCpltCallback\(\)\*](#)

### 54.2.4 Peripheral Control functions

This subsection provides a set of functions allowing to control the UART.

- [\*HAL\\_MultiProcessor\\_EnableMuteMode\(\)\*](#) API enables mute mode
- [\*HAL\\_MultiProcessor\\_DisableMuteMode\(\)\*](#) API disables mute mode
- [\*HAL\\_MultiProcessor\\_EnterMuteMode\(\)\*](#) API enters mute mode
- [\*HAL\\_HalfDuplex\\_EnableTransmitter\(\)\*](#) API disables receiver and enables transmitter
- [\*HAL\\_HalfDuplex\\_EnableReceiver\(\)\*](#) API disables transmitter and enables receiver
- [\*HAL\\_LIN\\_SendBreak\(\)\*](#) API transmits the break characters

This section contains the following APIs:

- [\*HAL\\_MultiProcessor\\_EnableMuteMode\(\)\*](#)
- [\*HAL\\_MultiProcessor\\_DisableMuteMode\(\)\*](#)
- [\*HAL\\_MultiProcessor\\_EnterMuteMode\(\)\*](#)
- [\*HAL\\_HalfDuplex\\_EnableTransmitter\(\)\*](#)
- [\*HAL\\_HalfDuplex\\_EnableReceiver\(\)\*](#)
- [\*HAL\\_LIN\\_SendBreak\(\)\*](#)

### 54.2.5 Peripheral State and Error functions

This subsection provides functions allowing to :

- Return the UART handle state.
- Return the UART handle error code

This section contains the following APIs:

- [\*\*HAL\\_UART\\_GetState\(\)\*\*](#)
- [\*\*HAL\\_UART\\_GetError\(\)\*\*](#)

#### 54.2.6 Detailed description of functions

##### **HAL\_UART\_Init**

|                      |                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_Init (UART_HandleTypeDef * huart)</b>                                                          |
| Function description | Initialize the UART mode according to the specified parameters in the UART_InitTypeDef and initialize the associated handle. |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                       |

##### **HAL\_HalfDuplex\_Init**

|                      |                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HalfDuplex_Init (UART_HandleTypeDef * huart)</b>                                                        |
| Function description | Initialize the half-duplex mode according to the specified parameters in the UART_InitTypeDef and creates the associated handle. |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                           |

##### **HAL\_LIN\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_LIN_Init (UART_HandleTypeDef * huart, uint32_t BreakDetectLength)</b>                                                                                                                                                                                                                                                                                                           |
| Function description | Initialize the LIN mode according to the specified parameters in the UART_InitTypeDef and creates the associated handle .                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>BreakDetectLength:</b> specifies the LIN break detection length. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <b>UART_LINBREAKDETECTLENGTH_10B</b> 10-bit break detection</li> <li>– <b>UART_LINBREAKDETECTLENGTH_11B</b> 11-bit break detection</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                   |

##### **HAL\_MultiProcessor\_Init**

|                      |                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_MultiProcessor_Init (UART_HandleTypeDef * huart, uint8_t Address, uint32_t WakeUpMethod)</b>                  |
| Function description | Initialize the multiprocessor mode according to the specified parameters in the UART_InitTypeDef and initialize the associated handle. |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li><b>huart:</b> UART handle.</li> <li><b>Address:</b> UART node address (4-, 6-, 7- or 8-bit long).</li> <li><b>WakeUpMethod:</b> specifies the UART wakeup method. This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– <b>UART_WAKEUPMETHOD_IDLELINE</b> WakeUp by an idle line detection</li> <li>– <b>UART_WAKEUPMETHOD_ADDRESSMARK</b> WakeUp by an address mark</li> </ul> </li> </ul>                        |
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes         | <ul style="list-style-type: none"> <li>If the user resorts to idle line detection wake up, the Address parameter is useless and ignored by the initialization function.</li> <li>If the user resorts to address mark wake up, the address length detection is configured by default to 4 bits only. For the UART to be able to manage 6-, 7- or 8-bit long addresses detection, the API <b>HAL_MultiProcessorEx_AddressLength_Set()</b> must be called after <b>HAL_MultiProcessor_Init()</b>.</li> </ul> |

### HAL\_UART\_DeInit

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_DeInit (UART_HandleTypeDef * huart)</b>        |
| Function description | DeInitialize the UART peripheral.                                            |
| Parameters           | <ul style="list-style-type: none"> <li><b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>         |

### HAL\_UART\_MspInit

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>void HAL_UART_MspInit (UART_HandleTypeDef * huart)</b>                    |
| Function description | Initialize the UART MSP.                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                |

### HAL\_UART\_MspDeInit

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>void HAL_UART_MspDeInit (UART_HandleTypeDef * huart)</b>                  |
| Function description | DeInitialize the UART MSP.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li><b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                |

### HAL\_UART\_Transmit

|                      |                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_Transmit (UART_HandleTypeDef * huart, uint8_t * pData, uint16_t Size, uint32_t Timeout)</b> |
| Function description | Send an amount of data in blocking mode.                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>huart:</b> UART handle.</li> </ul>                                              |

- **pData:** Pointer to data buffer.
- **Size:** Amount of data to be sent.
- **Timeout:** Timeout duration.
- **HAL:** status

Return values

**HAL\_UART\_Receive**

|                      |                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_Receive</b><br><b>(UART_HandleTypeDef * huart, uint8_t * pData, uint16_t Size,</b><br><b>uint32_t Timeout)</b>                                                                                        |
| Function description | Receive an amount of data in blocking mode.                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be received.</li> <li>• <b>Timeout:</b> Timeout duration.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                |

**HAL\_UART\_Transmit\_IT**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_Transmit_IT</b><br><b>(UART_HandleTypeDef * huart, uint8_t * pData, uint16_t Size)</b>                                                               |
| Function description | Send an amount of data in interrupt mode.                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be sent.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                               |

**HAL\_UART\_Receive\_IT**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_Receive_IT</b><br><b>(UART_HandleTypeDef * huart, uint8_t * pData, uint16_t Size)</b>                                                                    |
| Function description | Receive an amount of data in interrupt mode.                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be received.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                   |

**HAL\_UART\_Transmit\_DMA**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_Transmit_DMA</b><br><b>(UART_HandleTypeDef * huart, uint8_t * pData, uint16_t Size)</b>                                                              |
| Function description | Send an amount of data in DMA mode.                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be sent.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                               |

- |       |                                                                                                                                                                                                                        |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes | <ul style="list-style-type: none"> <li>• This function starts a DMA transfer in interrupt mode meaning that DMA half transfer complete, DMA transfer complete and DMA transfer error interrupts are enabled</li> </ul> |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### **HAL\_UART\_Receive\_DMA**

|                      |                                                                                                                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_Receive_DMA<br/>(UART_HandleTypeDef * huart, uint8_t * pData, uint16_t Size)</b>                                                                                                                                                                                                                        |
| Function description | Receive an amount of data in DMA mode.                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>pData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be received.</li> </ul>                                                                                                                                                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                |
| Notes                | <ul style="list-style-type: none"> <li>• When the UART parity is enabled (PCE = 1), the received data contain the parity bit (MSB position).</li> <li>• This function starts a DMA transfer in interrupt mode meaning that DMA half transfer complete, DMA transfer complete and DMA transfer error interrupts are enabled</li> </ul> |

### **HAL\_UART\_DMAPause**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_DMAPause<br/>(UART_HandleTypeDef * huart)</b>    |
| Function description | Pause the DMA Transfer.                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>         |

### **HAL\_UART\_DMAResume**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_DMAResume<br/>(UART_HandleTypeDef * huart)</b>   |
| Function description | Resume the DMA Transfer.                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>         |

### **HAL\_UART\_DMAStop**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_DMAStop<br/>(UART_HandleTypeDef * huart)</b>     |
| Function description | Stop the DMA Transfer.                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>         |

**HAL\_UART\_Abort**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_Abort<br/>(UART_HandleTypeDef * huart)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Abort ongoing transfers (blocking mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable UART Interrupts (Tx and Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort (in case of transfer in DMA mode)Set handle State to READY</li> <li>• This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.</li> </ul> |

**HAL\_UART\_AbortTransmit**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_AbortTransmit<br/>(UART_HandleTypeDef * huart)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Abort ongoing Transmit transfer (blocking mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Tx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable UART Interrupts (Tx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort (in case of transfer in DMA mode)Set handle State to READY</li> <li>• This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.</li> </ul> |

**HAL\_UART\_AbortReceive**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_AbortReceive<br/>(UART_HandleTypeDef * huart)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description | Abort ongoing Receive transfer (blocking mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Rx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable UART Interrupts (Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort (in case of transfer in DMA mode)Set handle State to READY</li> <li>• This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.</li> </ul> |

**HAL\_UART\_Abort\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_Abort_IT<br/>(UART_HandleTypeDef * huart)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Abort ongoing transfers (Interrupt mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable UART Interrupts (Tx and Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort_IT (in case of transfer in DMA mode)Set handle State to READYAt abort completion, call user abort complete callback</li> <li>• This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).</li> </ul> |

**HAL\_UART\_AbortTransmit\_IT**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_AbortTransmit_IT<br/>(UART_HandleTypeDef * huart)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Abort ongoing Transmit transfer (Interrupt mode).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Tx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable UART Interrupts (Tx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL_DMA_Abort_IT (in case of transfer in DMA mode)Set handle State to READYAt abort completion, call user abort complete callback</li> <li>• This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).</li> </ul> |

**HAL\_UART\_AbortReceive\_IT**

|                      |                                                                                                                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UART_AbortReceive_IT<br/>(UART_HandleTypeDef * huart)</b>                                                                                                                                                                                                         |
| Function description | Abort ongoing Receive transfer (Interrupt mode).                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                                                                                                                                                                             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This procedure could be used for aborting any ongoing Rx transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable UART Interrupts (Rx)Disable the DMA transfer in the peripheral register (if</li> </ul> |

- enabled) Abort DMA transfer by calling HAL\_DMA\_Abort\_IT  
 (in case of transfer in DMA mode) Set handle State to  
 READY At abort completion, call user abort complete callback
- This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).

### **HAL\_UART\_IRQHandler**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>void HAL_UART_IRQHandler (UART_HandleTypeDef * huart)</b>                 |
| Function description | Handle UART interrupt request.                                               |
| Parameters           | <ul style="list-style-type: none"> <li><b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                |

### **HAL\_UART\_TxCpltCallback**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>void HAL_UART_TxCpltCallback (UART_HandleTypeDef * huart)</b>             |
| Function description | Tx Transfer completed callback.                                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                |

### **HAL\_UART\_TxHalfCpltCallback**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>void HAL_UART_TxHalfCpltCallback (UART_HandleTypeDef * huart)</b>         |
| Function description | Tx Half Transfer completed callback.                                         |
| Parameters           | <ul style="list-style-type: none"> <li><b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                |

### **HAL\_UART\_RxCpltCallback**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>void HAL_UART_RxCpltCallback (UART_HandleTypeDef * huart)</b>             |
| Function description | Rx Transfer completed callback.                                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                |

### **HAL\_UART\_RxHalfCpltCallback**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>void HAL_UART_RxHalfCpltCallback (UART_HandleTypeDef * huart)</b>         |
| Function description | Rx Half Transfer completed callback.                                         |
| Parameters           | <ul style="list-style-type: none"> <li><b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                |

**HAL\_UART\_ErrorCallback**

Function name      **void HAL\_UART\_ErrorCallback (UART\_HandleTypeDef \* huart)**

Function description      UART error callback.

Parameters      •    **huart:** UART handle.

Return values      •    **None**

**HAL\_UART\_AbortCpltCallback**

Function name      **void HAL\_UART\_AbortCpltCallback (UART\_HandleTypeDef \* huart)**

Function description      UART Abort Complete callback.

Parameters      •    **huart:** UART handle.

Return values      •    **None**

**HAL\_UART\_AbortTransmitCpltCallback**

Function name      **void HAL\_UART\_AbortTransmitCpltCallback (UART\_HandleTypeDef \* huart)**

Function description      UART Abort Complete callback.

Parameters      •    **huart:** UART handle.

Return values      •    **None**

**HAL\_UART\_AbortReceiveCpltCallback**

Function name      **void HAL\_UART\_AbortReceiveCpltCallback (UART\_HandleTypeDef \* huart)**

Function description      UART Abort Receive Complete callback.

Parameters      •    **huart:** UART handle.

Return values      •    **None**

**HAL\_MultiProcessor\_EnableMuteMode**

Function name      **HAL\_StatusTypeDef HAL\_MultiProcessor\_EnableMuteMode (UART\_HandleTypeDef \* huart)**

Function description      Enable UART in mute mode (does not mean UART enters mute mode; to enter mute mode, HAL\_MultiProcessor\_EnterMuteMode() API must be called).

Parameters      •    **huart:** UART handle.

Return values      •    **HAL:** status

**HAL\_MultiProcessor\_DisableMuteMode**

|                      |                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_MultiProcessor_DisableMuteMode<br/>(UART_HandleTypeDef * huart)</b>                                       |
| Function description | Disable UART mute mode (does not mean the UART actually exits mute mode as it may not have been in mute mode at this very moment). |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                             |

**HAL\_MultiProcessor\_EnterMuteMode**

|                      |                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_MultiProcessor_EnterMuteMode<br/>(UART_HandleTypeDef * huart)</b>                                                        |
| Function description | Enter UART mute mode (means UART actually enters mute mode).                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• To exit from mute mode, HAL_MultiProcessor_DisableMuteMode() API must be called.</li> </ul> |

**HAL\_HalfDuplex\_EnableTransmitter**

|                      |                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HalfDuplex_EnableTransmitter<br/>(UART_HandleTypeDef * huart)</b> |
| Function description | Enable the UART transmitter and disable the UART receiver.                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                     |

**HAL\_HalfDuplex\_EnableReceiver**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_HalfDuplex_EnableReceiver<br/>(UART_HandleTypeDef * huart)</b> |
| Function description | Enable the UART receiver and disable the UART transmitter.                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status.</li> </ul>                 |

**HAL\_LIN\_SendBreak**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_LIN_SendBreak<br/>(UART_HandleTypeDef * huart)</b>    |
| Function description | Transmit break characters.                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>         |

**HAL\_UART\_GetState**

|                      |                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_UART_StateTypeDef HAL_UART_GetState (UART_HandleTypeDef * huart)</b>                                                                                                   |
| Function description | Return the UART handle state.                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"><li>• <b>huart:</b> Pointer to a UART_HandleTypeDef structure that contains the configuration information for the specified UART.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> state</li></ul>                                                                                                           |

**HAL\_UART\_GetError**

|                      |                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t HAL_UART_GetError (UART_HandleTypeDef * huart)</b>                                                                                                                |
| Function description | Return the UART handle error code.                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"><li>• <b>huart:</b> Pointer to a UART_HandleTypeDef structure that contains the configuration information for the specified UART.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>UART:</b> Error Code</li></ul>                                                                                                     |

**UART\_SetConfig**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef UART_SetConfig (UART_HandleTypeDef * huart)</b>         |
| Function description | Configure the UART peripheral.                                               |
| Parameters           | <ul style="list-style-type: none"><li>• <b>huart:</b> UART handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>         |

**UART\_AdvFeatureConfig**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>void UART_AdvFeatureConfig (UART_HandleTypeDef * huart)</b>               |
| Function description | Configure the UART peripheral advanced features.                             |
| Parameters           | <ul style="list-style-type: none"><li>• <b>huart:</b> UART handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                |

**UART\_CheckIdleState**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef UART_CheckIdleState (UART_HandleTypeDef * huart)</b>    |
| Function description | Check the UART Idle State.                                                   |
| Parameters           | <ul style="list-style-type: none"><li>• <b>huart:</b> UART handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>HAL:</b> status</li></ul>         |

**UART\_WaitOnFlagUntilTimeout**

|                      |                                                                                                                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef</b> <b>UART_WaitOnFlagUntilTimeout</b><br><b>(UART_HandleTypeDef * huart, uint32_t Flag, FlagStatus Status, uint32_t Tickstart, uint32_t Timeout)</b>                                                                                                             |
| Function description | Handle UART Communication Timeout.                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>Flag:</b> Specifies the UART flag to check</li> <li>• <b>Status:</b> Flag status (SET or RESET)</li> <li>• <b>Tickstart:</b> Tick start value</li> <li>• <b>Timeout:</b> Timeout duration</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                 |

**UART\_Transmit\_IT**

|                      |                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef</b> <b>UART_Transmit_IT</b><br><b>(UART_HandleTypeDef * huart)</b>                                                                         |
| Function description | Send an amount of data in interrupt mode.                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                                                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• Function is called under interruption only, once interruptions have been enabled by HAL_UART_Transmit_IT().</li> </ul> |

**UART\_EndTransmit\_IT**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef</b> <b>UART_EndTransmit_IT</b><br><b>(UART_HandleTypeDef * huart)</b>                                                                                             |
| Function description | Wrap up transmission in non-blocking mode.                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> pointer to a UART_HandleTypeDef structure that contains the configuration information for the specified UART module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                 |

**UART\_Receive\_IT**

|                      |                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef</b> <b>UART_Receive_IT</b><br><b>(UART_HandleTypeDef * huart)</b>                                                                        |
| Function description | Receive an amount of data in interrupt mode.                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                                                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• Function is called under interruption only, once interruptions have been enabled by HAL_UART_Receive_IT()</li> </ul> |

**UART\_Wakeup\_AddressConfig**

|                      |                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void UART_Wakeup_AddressConfig (UART_HandleTypeDef * huart, UART_WakeUpTypeDef WakeUpSelection)</b>                                                    |
| Function description | Initialize the UART wake-up from stop mode parameters when triggered by address detection.                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>WakeUpSelection:</b> UART wake up from stop mode parameters.</li> </ul> |
| Return values        | • <b>None</b>                                                                                                                                             |

**54.3 UART Firmware driver defines****54.3.1 UART*****UART Advanced Feature Initialization Type***

|                                        |                                          |
|----------------------------------------|------------------------------------------|
| UART_ADVFEATURE_NO_INIT                | No advanced feature initialization       |
| UART_ADVFEATURE_TXINVERT_INIT          | TX pin active level inversion            |
| UART_ADVFEATURE_RXINVERT_INIT          | RX pin active level inversion            |
| UART_ADVFEATURE_DATAINVERT_INIT        | Binary data inversion                    |
| UART_ADVFEATURE_SWAP_INIT              | TX/RX pins swap                          |
| UART_ADVFEATURE_RXOVERRUNDISABLE_INIT  | RX overrun disable                       |
| UART_ADVFEATURE_DMADISABLEONERROR_INIT | DMA disable on Reception Error           |
| UART_ADVFEATURE_AUTOBAUDRATE_INIT      | Auto Baud rate detection initialization  |
| UART_ADVFEATURE_MSBFIRST_INIT          | Most significant bit sent/received first |

***UART Advanced Feature Auto BaudRate Enable***

|                                      |                                     |
|--------------------------------------|-------------------------------------|
| UART_ADVFEATURE_AUTOBAUDRATE_DISABLE | RX Auto Baud rate detection enable  |
| UART_ADVFEATURE_AUTOBAUDRATE_ENABLE  | RX Auto Baud rate detection disable |

***UART Advanced Feature AutoBaud Rate Mode***

|                                            |                                                  |
|--------------------------------------------|--------------------------------------------------|
| UART_ADVFEATURE_AUTOBAUDRATE_ONSTARTBIT    | Auto Baud rate detection on start bit            |
| UART_ADVFEATURE_AUTOBAUDRATE_ONFALLINGEDGE | Auto Baud rate detection on falling edge         |
| UART_ADVFEATURE_AUTOBAUDRATE_ON0X7FFFRAME  | Auto Baud rate detection on 0x7F frame detection |
| UART_ADVFEATURE_AUTOBAUDRATE_ON0X55FRAME   | Auto Baud rate detection on 0x55 frame detection |

***UART Driver Enable Assertion Time LSB Position In CR1 Register***

|                               |                                       |
|-------------------------------|---------------------------------------|
| UART_CR1_DEAT_ADDRESS_LSB_POS | UART Driver Enable assertion time LSB |
|-------------------------------|---------------------------------------|

position in CR1 register

**UART Driver Enable DeAssertion Time LSB Position In CR1 Register**

`UART_CR1_DEDT_ADDRESS_LSB_POS`    UART Driver Enable de-assertion time LSB position in CR1 register

**UART Address-matching LSB Position In CR2 Register**

`UART_CR2_ADDRESS_LSB_POS`    UART address-matching LSB position in CR2 register

**UART Advanced Feature Binary Data Inversion**

`UART_ADVFEATURE_DATAINV_DISABLE`    Binary data inversion disable

`UART_ADVFEATURE_DATAINV_ENABLE`    Binary data inversion enable

**UART Advanced Feature DMA Disable On Rx Error**

`UART_ADVFEATURE_DMA_ENABLEONRXERROR`    DMA enable on Reception Error

`UART_ADVFEATURE_DMA_DISABLEONRXERROR`    DMA disable on Reception Error

**UART DMA Rx**

`UART_DMA_RX_DISABLE`    UART DMA RX disabled

`UART_DMA_RX_ENABLE`    UART DMA RX enabled

**UART DMA Tx**

`UART_DMA_TX_DISABLE`    UART DMA TX disabled

`UART_DMA_TX_ENABLE`    UART DMA TX enabled

**UART DriverEnable Polarity**

`UART_DE_POLARITY_HIGH`    Driver enable signal is active high

`UART_DE_POLARITY_LOW`    Driver enable signal is active low

**UART Error**

`HAL_UART_ERROR_NONE`    No error

`HAL_UART_ERROR_PE`    Parity error

`HAL_UART_ERROR_NE`    Noise error

`HAL_UART_ERROR_FE`    frame error

`HAL_UART_ERROR_ORE`    Overrun error

`HAL_UART_ERROR_DMA`    DMA transfer error

`HAL_UART_ERROR_BUSY`    Busy Error

**UART Exported Macros**

`_HAL_UART_RESET_HANDLE_STA`    **Description:**  
TE

- Reset UART handle states.

**Parameters:**

- `_HANDLE_`: UART handle.

**Return value:**

- None

|                                          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>__HAL_UART_FLUSH_DRREGISTER</code> | <p><b>Description:</b></p> <ul style="list-style-type: none"><li>Flush the UART Data registers.</li></ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"><li><code>__HANDLE__</code>: specifies the UART Handle.</li></ul> <p><b>Return value:</b></p> <ul style="list-style-type: none"><li>None</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <code>__HAL_UART_CLEAR_FLAG</code>       | <p><b>Description:</b></p> <ul style="list-style-type: none"><li>Clear the specified UART pending flag.</li></ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"><li><code>__HANDLE__</code>: specifies the UART Handle.</li><li><code>__FLAG__</code>: specifies the flag to check. This parameter can be any combination of the following values:<ul style="list-style-type: none"><li><code>UART_CLEAR_PEF</code> Parity Error Clear Flag</li><li><code>UART_CLEAR_FEF</code> Framing Error Clear Flag</li><li><code>UART_CLEAR_NEF</code> Noise detected Clear Flag</li><li><code>UART_CLEAR_OREF</code> Overrun Error Clear Flag</li><li><code>UART_CLEAR_IDLEF</code> IDLE line detected Clear Flag</li><li><code>UART_CLEAR_TCF</code> Transmission Complete Clear Flag</li><li><code>UART_CLEAR_LBDF</code> LIN Break Detection Clear Flag (not available on all devices)</li><li><code>UART_CLEAR_CTSF</code> CTS Interrupt Clear Flag</li><li><code>UART_CLEAR_RTOF</code> Receiver Time Out Clear Flag</li><li><code>UART_CLEAR_EOBF</code> End Of Block Clear Flag (not available on all devices)</li><li><code>UART_CLEAR_CMF</code> Character Match Clear Flag</li><li><code>UART_CLEAR_WUF</code> Wake Up from stop mode Clear Flag (not available on all devices)</li></ul></li></ul> <p><b>Return value:</b></p> <ul style="list-style-type: none"><li>None</li></ul> |
| <code>__HAL_UART_CLEAR_PEFLAG</code>     | <p><b>Description:</b></p> <ul style="list-style-type: none"><li>Clear the UART PE pending flag.</li></ul> <p><b>Parameters:</b></p> <ul style="list-style-type: none"><li><code>__HANDLE__</code>: specifies the UART Handle.</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

`__HAL_UART_CLEAR_FEFLAG`

**Return value:**

- None

**Description:**

- Clear the UART FE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

`__HAL_UART_CLEAR_NEFLAG`

**Description:**

- Clear the UART NE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

`__HAL_UART_CLEAR_OREFLAG`

**Description:**

- Clear the UART ORE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

`__HAL_UART_CLEAR_IDLEFLAG`

**Description:**

- Clear the UART IDLE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

`__HAL_UART_GET_FLAG`

**Description:**

- Check whether the specified UART flag is set or not.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.
- `__FLAG__`: specifies the flag to check. This parameter can be one of the following values:
  - `UART_FLAG_RXACK` Receive enable acknowledge flag
  - `UART_FLAG_TEACK` Transmit enable acknowledge flag
  - `UART_FLAG_WUF` Wake up from stop

- mode flag
- UART\_FLAG\_RWU Receiver wake up flag
- UART\_FLAG\_SBKF Send Break flag
- UART\_FLAG\_CMF Character match flag
- UART\_FLAG\_BUSY Busy flag
- UART\_FLAG\_ABRF Auto Baud rate detection flag
- UART\_FLAG\_ABRE Auto Baud rate detection error flag
- UART\_FLAG\_EOBF End of block flag
- UART\_FLAG\_RTOF Receiver timeout flag
- UART\_FLAG\_CTS CTS Change flag (not available for UART4 and UART5)
- UART\_FLAG\_LBDF LIN Break detection flag
- UART\_FLAG\_TXE Transmit data register empty flag
- UART\_FLAG\_TC Transmission Complete flag
- UART\_FLAG\_RXNE Receive data register not empty flag
- UART\_FLAG\_IDLE Idle Line detection flag
- UART\_FLAG\_ORE Overrun Error flag
- UART\_FLAG\_NE Noise Error flag
- UART\_FLAG\_FE Framing Error flag
- UART\_FLAG\_PE Parity Error flag

**Return value:**

- The new state of \_\_FLAG\_\_ (TRUE or FALSE).

**\_\_HAL\_UART\_ENABLE\_IT****Description:**

- Enable the specified UART interrupt.

**Parameters:**

- \_\_HANDLE\_\_: specifies the UART Handle.
- \_\_INTERRUPT\_\_: specifies the UART interrupt source to enable. This parameter can be one of the following values:
  - UART\_IT\_WUF Wakeup from stop mode interrupt
  - UART\_IT\_CM Character match interrupt
  - UART\_IT\_CTS CTS change interrupt
  - UART\_IT\_LBD LIN Break detection interrupt
  - UART\_IT\_TXE Transmit Data Register empty interrupt
  - UART\_IT\_TC Transmission complete interrupt

- UART\_IT\_RXNE Receive Data register not empty interrupt
- UART\_IT\_IDLE Idle line detection interrupt
- UART\_IT\_PE Parity Error interrupt
- UART\_IT\_ERR Error interrupt (Frame error, noise error, overrun error)

**Return value:**

- None

**\_HAL\_UART\_DISABLE\_IT**

- Disable the specified UART interrupt.

**Parameters:**

- HANDLE: specifies the UART Handle.
- INTERRUPT: specifies the UART interrupt source to disable. This parameter can be one of the following values:
  - UART\_IT\_WUF Wakeup from stop mode interrupt
  - UART\_IT\_CM Character match interrupt
  - UART\_IT\_CTS CTS change interrupt
  - UART\_IT\_LBD LIN Break detection interrupt
  - UART\_IT\_TXE Transmit Data Register empty interrupt
  - UART\_IT\_TC Transmission complete interrupt
  - UART\_IT\_RXNE Receive Data register not empty interrupt
  - UART\_IT\_IDLE Idle line detection interrupt
  - UART\_IT\_PE Parity Error interrupt
  - UART\_IT\_ERR Error interrupt (Frame error, noise error, overrun error)

**Return value:**

- None

**\_HAL\_UART\_GET\_IT**

- Check whether the specified UART interrupt has occurred or not.

**Parameters:**

- HANDLE: specifies the UART Handle.
- IT: specifies the UART interrupt to check. This parameter can be one of the following values:
  - UART\_IT\_WUF Wakeup from stop mode interrupt
  - UART\_IT\_CM Character match interrupt

- UART\_IT\_CTS CTS change interrupt (not available for UART4 and UART5)
- UART\_IT\_LBD LIN Break detection interrupt
- UART\_IT\_TXE Transmit Data Register empty interrupt
- UART\_IT\_TC Transmission complete interrupt
- UART\_IT\_RXNE Receive Data register not empty interrupt
- UART\_IT\_IDLE Idle line detection interrupt
- UART\_IT\_ORE Overrun Error interrupt
- UART\_IT\_NE Noise Error interrupt
- UART\_IT\_FE Framing Error interrupt
- UART\_IT\_PE Parity Error interrupt

**Return value:**

- The: new state of `__IT__` (TRUE or FALSE).

**`__HAL_UART_GET_IT_SOURCE`****Description:**

- Check whether the specified UART interrupt source is enabled or not.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.
- `__IT__`: specifies the UART interrupt source to check. This parameter can be one of the following values:
  - UART\_IT\_WUF Wakeup from stop mode interrupt
  - UART\_IT\_CM Character match interrupt
  - UART\_IT\_CTS CTS change interrupt (not available for UART4 and UART5)
  - UART\_IT\_LBD LIN Break detection interrupt
  - UART\_IT\_TXE Transmit Data Register empty interrupt
  - UART\_IT\_TC Transmission complete interrupt
  - UART\_IT\_RXNE Receive Data register not empty interrupt
  - UART\_IT\_IDLE Idle line detection interrupt
  - UART\_IT\_ERR Error interrupt (Frame error, noise error, overrun error)
  - UART\_IT\_PE Parity Error interrupt

**Return value:**

- The: new state of `__IT__` (TRUE or FALSE).

[\\_\\_HAL\\_UART\\_CLEAR\\_IT](#)**Description:**

- Clear the specified UART ISR flag, in setting the proper ICR register flag.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the UART Handle.
- [\\_\\_IT\\_CLEAR\\_\\_](#): specifies the interrupt clear register flag that needs to be set to clear the corresponding interrupt This parameter can be one of the following values:
  - [UART\\_CLEAR\\_PEF](#) Parity Error Clear Flag
  - [UART\\_CLEAR\\_FEF](#) Framing Error Clear Flag
  - [UART\\_CLEAR\\_NEF](#) Noise detected Clear Flag
  - [UART\\_CLEAR\\_OREF](#) Overrun Error Clear Flag
  - [UART\\_CLEAR\\_IDLEF](#) IDLE line detected Clear Flag
  - [UART\\_CLEAR\\_TCF](#) Transmission Complete Clear Flag
  - [UART\\_CLEAR\\_LBDF](#) LIN Break Detection Clear Flag
  - [UART\\_CLEAR\\_CTSF](#) CTS Interrupt Clear Flag
  - [UART\\_CLEAR\\_RTOF](#) Receiver Time Out Clear Flag
  - [UART\\_CLEAR\\_EOBF](#) End Of Block Clear Flag
  - [UART\\_CLEAR\\_CMF](#) Character Match Clear Flag
  - [UART\\_CLEAR\\_WUF](#) Wake Up from stop mode Clear Flag

**Return value:**

- None

[\\_\\_HAL\\_UART\\_SEND\\_REQ](#)**Description:**

- Set a specific UART request flag.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the UART Handle.
- [\\_\\_REQ\\_\\_](#): specifies the request flag to set This parameter can be one of the following values:
  - [UART\\_AUTOBAUD\\_REQUEST](#) Auto-Baud Rate Request
  - [UART\\_SENDBREAK\\_REQUEST](#) Send Break Request
  - [UART\\_MUTE\\_MODE\\_REQUEST](#) Mute Mode Request
  - [UART\\_RXDATA\\_FLUSH\\_REQUEST](#)

- Receive Data flush Request
- UART\_TXDATA\_FLUSH\_REQUEST
- Transmit data flush Request

**Return value:**

- None

`__HAL_UART_ONE_BIT_SAMPLE_ENABLE`

**Description:**

- Enable the UART one bit sample method.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

`__HAL_UART_ONE_BIT_SAMPLE_DISABLE`

**Description:**

- Disable the UART one bit sample method.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

`__HAL_UART_ENABLE`

**Description:**

- Enable UART.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

`__HAL_UART_DISABLE`

**Description:**

- Disable UART.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

`__HAL_UART_HWCONTROL_CTS_ENABLE`

**Description:**

- Enable CTS flow control.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

**Notes:**

- This macro allows to enable CTS hardware

flow control for a given UART instance, without need to call HAL\_UART\_Init() function. As involving direct access to UART registers, usage of this macro should be fully endorsed by user. As macro is expected to be used for modifying CTS Hw flow control feature activation, without need for USART instance Deinit/Init, following conditions for macro call should be fulfilled : UART instance should have already been initialised (through call of HAL\_UART\_Init()) macro could only be called when corresponding UART instance is disabled (i.e.

`_HAL_UART_DISABLE(__HANDLE__)`  
and should be followed by an Enable macro (i.e.  
`_HAL_UART_ENABLE(__HANDLE__)`).

### `_HAL_UART_HWCONTROL_CTS_DISABLE`

#### **Description:**

- Disable CTS flow control.

#### **Parameters:**

- `__HANDLE__`: specifies the UART Handle.

#### **Return value:**

- None

#### **Notes:**

• This macro allows to disable CTS hardware flow control for a given UART instance, without need to call HAL\_UART\_Init() function. As involving direct access to UART registers, usage of this macro should be fully endorsed by user. As macro is expected to be used for modifying CTS Hw flow control feature activation, without need for USART instance Deinit/Init, following conditions for macro call should be fulfilled : UART instance should have already been initialised (through call of HAL\_UART\_Init()) macro could only be called when corresponding UART instance is disabled (i.e.  
`_HAL_UART_DISABLE(__HANDLE__)`  
and should be followed by an Enable macro (i.e.  
`_HAL_UART_ENABLE(__HANDLE__)`).

### `_HAL_UART_HWCONTROL_RTS_ENABLE`

#### **Description:**

- Enable RTS flow control.

#### **Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

**Notes:**

- This macro allows to enable RTS hardware flow control for a given UART instance, without need to call HAL\_UART\_Init() function. As involving direct access to UART registers, usage of this macro should be fully endorsed by user. As macro is expected to be used for modifying RTS Hw flow control feature activation, without need for USART instance Deinit/Init, following conditions for macro call should be fulfilled : UART instance should have already been initialised (through call of HAL\_UART\_Init()) macro could only be called when corresponding UART instance is disabled (i.e. `__HAL_UART_DISABLE(__HANDLE__)`) and should be followed by an Enable macro (i.e. `__HAL_UART_ENABLE(__HANDLE__)`).

**`__HAL_UART_HWCONTROL_RTS_DISABLE`****Description:**

- Disable RTS flow control.

**Parameters:**

- `__HANDLE__`: specifies the UART Handle.

**Return value:**

- None

**Notes:**

- This macro allows to disable RTS hardware flow control for a given UART instance, without need to call HAL\_UART\_Init() function. As involving direct access to UART registers, usage of this macro should be fully endorsed by user. As macro is expected to be used for modifying RTS Hw flow control feature activation, without need for USART instance Deinit/Init, following conditions for macro call should be fulfilled : UART instance should have already been initialised (through call of HAL\_UART\_Init()) macro could only be called when corresponding UART instance is disabled (i.e. `__HAL_UART_DISABLE(__HANDLE__)`) and should be followed by an Enable macro (i.e. `__HAL_UART_ENABLE(__HANDLE__)`).

***UART Status Flags***

|                 |                                           |
|-----------------|-------------------------------------------|
| UART_FLAG_RXACK | UART receive enable acknowledge flag      |
| UART_FLAG_TEACK | UART transmit enable acknowledge flag     |
| UART_FLAG_WUF   | UART wake-up from stop mode flag          |
| UART_FLAG_RWU   | UART receiver wake-up from mute mode flag |
| UART_FLAG_SBKF  | UART send break flag                      |
| UART_FLAG_CMF   | UART character match flag                 |
| UART_FLAG_BUSY  | UART busy flag                            |
| UART_FLAG_ABRF  | UART auto Baud rate flag                  |
| UART_FLAG_ABRE  | UART auto Baud rate error                 |
| UART_FLAG_EOBF  | UART end of block flag                    |
| UART_FLAG_RTOF  | UART receiver timeout flag                |
| UART_FLAG_CTS   | UART clear to send flag                   |
| UART_FLAG_CTSIF | UART clear to send interrupt flag         |
| UART_FLAG_LBDF  | UART LIN break detection flag             |
| UART_FLAG_TXE   | UART transmit data register empty         |
| UART_FLAG_TC    | UART transmission complete                |
| UART_FLAG_RXNE  | UART read data register not empty         |
| UART_FLAG_IDLE  | UART idle flag                            |
| UART_FLAG_ORE   | UART overrun error                        |
| UART_FLAG_NE    | UART noise error                          |
| UART_FLAG_FE    | UART frame error                          |
| UART_FLAG_PE    | UART parity error                         |

***UART Half Duplex Selection***

|                          |                           |
|--------------------------|---------------------------|
| UART_HALF_DUPLEX_DISABLE | UART half-duplex disabled |
| UART_HALF_DUPLEX_ENABLE  | UART half-duplex enabled  |

***UART Hardware Flow Control***

|                        |                           |
|------------------------|---------------------------|
| UART_HWCONTROL_NONE    | No hardware control       |
| UART_HWCONTROL_RTS     | Request To Send           |
| UART_HWCONTROL_CTS     | Clear To Send             |
| UART_HWCONTROL_RTS_CTS | Request and Clear To Send |

***UART Interruptions Flag Mask***

|              |                               |
|--------------|-------------------------------|
| UART_IT_MASK | UART interruptions flags mask |
|--------------|-------------------------------|

***UART Interrupts Definition***

|             |                                                |
|-------------|------------------------------------------------|
| UART_IT_PE  | UART parity error interruption                 |
| UART_IT_TXE | UART transmit data register empty interruption |

---

|              |                                                |
|--------------|------------------------------------------------|
| UART_IT_TC   | UART transmission complete interruption        |
| UART_IT_RXNE | UART read data register not empty interruption |
| UART_IT_IDLE | UART idle interruption                         |
| UART_IT_LBD  | UART LIN break detection interruption          |
| UART_IT_CTS  | UART CTS interruption                          |
| UART_IT_CM   | UART character match interruption              |
| UART_IT_WUF  | UART wake-up from stop mode interruption       |
| UART_IT_ERR  | UART error interruption                        |
| UART_IT_ORE  | UART overrun error interruption                |
| UART_IT_NE   | UART noise error interruption                  |
| UART_IT_FE   | UART frame error interruption                  |

***UART Interruption Clear Flags***

|                  |                                   |
|------------------|-----------------------------------|
| UART_CLEAR_PEF   | Parity Error Clear Flag           |
| UART_CLEAR_FEF   | Framing Error Clear Flag          |
| UART_CLEAR_NEF   | Noise detected Clear Flag         |
| UART_CLEAR_OREF  | Overrun Error Clear Flag          |
| UART_CLEAR_IDLEF | IDLE line detected Clear Flag     |
| UART_CLEAR_TCF   | Transmission Complete Clear Flag  |
| UART_CLEAR_LBDF  | LIN Break Detection Clear Flag    |
| UART_CLEAR_CTSF  | CTS Interrupt Clear Flag          |
| UART_CLEAR_RTOF  | Receiver Time Out Clear Flag      |
| UART_CLEAR_EOBF  | End Of Block Clear Flag           |
| UART_CLEAR_CMF   | Character Match Clear Flag        |
| UART_CLEAR_WUF   | Wake Up from stop mode Clear Flag |

***UART Local Interconnection Network mode***

|                  |                                    |
|------------------|------------------------------------|
| UART_LIN_DISABLE | Local Interconnect Network disable |
| UART_LIN_ENABLE  | Local Interconnect Network enable  |

***UART LIN Break Detection***

|                               |                                   |
|-------------------------------|-----------------------------------|
| UART_LINBREAKDETECTLENGTH_10B | LIN 10-bit break detection length |
| UART_LINBREAKDETECTLENGTH_11B | LIN 11-bit break detection length |

***UART Transfer Mode***

|                 |                |
|-----------------|----------------|
| UART_MODE_RX    | RX mode        |
| UART_MODE_TX    | TX mode        |
| UART_MODE_TX_RX | RX and TX mode |

***UART Advanced Feature MSB First***

|                                  |                                                  |
|----------------------------------|--------------------------------------------------|
| UART_ADVFEATURE_MSBFIRST_DISABLE | Most significant bit sent/received first disable |
|----------------------------------|--------------------------------------------------|

|                                 |                                                 |
|---------------------------------|-------------------------------------------------|
| UART_ADVFEATURE_MSBFIRST_ENABLE | Most significant bit sent/received first enable |
|---------------------------------|-------------------------------------------------|

**UART Advanced Feature Mute Mode Enable**

|                                  |                        |
|----------------------------------|------------------------|
| UART_ADVFEATURE_MUTEMODE_DISABLE | UART mute mode disable |
|----------------------------------|------------------------|

|                                 |                       |
|---------------------------------|-----------------------|
| UART_ADVFEATURE_MUTEMODE_ENABLE | UART mute mode enable |
|---------------------------------|-----------------------|

**UART One Bit Sampling Method**

|                             |                          |
|-----------------------------|--------------------------|
| UART_ONE_BIT_SAMPLE_DISABLE | One-bit sampling disable |
|-----------------------------|--------------------------|

|                            |                         |
|----------------------------|-------------------------|
| UART_ONE_BIT_SAMPLE_ENABLE | One-bit sampling enable |
|----------------------------|-------------------------|

**UART Advanced Feature Overrun Disable**

|                                |                   |
|--------------------------------|-------------------|
| UART_ADVFEATURE_OVERRUN_ENABLE | RX overrun enable |
|--------------------------------|-------------------|

|                                 |                    |
|---------------------------------|--------------------|
| UART_ADVFEATURE_OVERRUN_DISABLE | RX overrun disable |
|---------------------------------|--------------------|

**UART Over Sampling**

|                      |                     |
|----------------------|---------------------|
| UART_OVERSAMPLING_16 | Oversampling by 16U |
|----------------------|---------------------|

|                     |                   |
|---------------------|-------------------|
| UART_OVERSAMPLING_8 | Oversampling by 8 |
|---------------------|-------------------|

**UART Parity**

|                  |           |
|------------------|-----------|
| UART_PARITY_NONE | No parity |
|------------------|-----------|

|                  |             |
|------------------|-------------|
| UART_PARITY EVEN | Even parity |
|------------------|-------------|

|                 |            |
|-----------------|------------|
| UART_PARITY ODD | Odd parity |
|-----------------|------------|

**UART Receiver TimeOut**

|                               |                               |
|-------------------------------|-------------------------------|
| UART_RECEIVER_TIMEOUT_DISABLE | UART receiver timeout disable |
|-------------------------------|-------------------------------|

|                              |                              |
|------------------------------|------------------------------|
| UART_RECEIVER_TIMEOUT_ENABLE | UART receiver timeout enable |
|------------------------------|------------------------------|

**UART Request Parameters**

|                       |                        |
|-----------------------|------------------------|
| UART_AUTOBAUD_REQUEST | Auto-Baud Rate Request |
|-----------------------|------------------------|

|                        |                    |
|------------------------|--------------------|
| UART_SENDBREAK_REQUEST | Send Break Request |
|------------------------|--------------------|

|                        |                   |
|------------------------|-------------------|
| UART_MUTE_MODE_REQUEST | Mute Mode Request |
|------------------------|-------------------|

|                           |                            |
|---------------------------|----------------------------|
| UART_RXDATA_FLUSH_REQUEST | Receive Data flush Request |
|---------------------------|----------------------------|

|                           |                             |
|---------------------------|-----------------------------|
| UART_TXDATA_FLUSH_REQUEST | Transmit data flush Request |
|---------------------------|-----------------------------|

**UART Advanced Feature RX Pin Active Level Inversion**

|                               |                                       |
|-------------------------------|---------------------------------------|
| UART_ADVFEATURE_RXINV_DISABLE | RX pin active level inversion disable |
|-------------------------------|---------------------------------------|

|                              |                                      |
|------------------------------|--------------------------------------|
| UART_ADVFEATURE_RXINV_ENABLE | RX pin active level inversion enable |
|------------------------------|--------------------------------------|

**UART Advanced Feature RX TX Pins Swap**

|                              |                         |
|------------------------------|-------------------------|
| UART_ADVFEATURE_SWAP_DISABLE | TX/RX pins swap disable |
|------------------------------|-------------------------|

|                             |                        |
|-----------------------------|------------------------|
| UART_ADVFEATURE_SWAP_ENABLE | TX/RX pins swap enable |
|-----------------------------|------------------------|

**UART State**

|                    |               |
|--------------------|---------------|
| UART_STATE_DISABLE | UART disabled |
|--------------------|---------------|

|                   |              |
|-------------------|--------------|
| UART_STATE_ENABLE | UART enabled |
|-------------------|--------------|

## *UART Number of Stop Bits*

|                                |                               |
|--------------------------------|-------------------------------|
| <code>UART_STOPBITS_0_5</code> | UART frame with 0.5 stop bit  |
| <code>UART_STOPBITS_1</code>   | UART frame with 1 stop bit    |
| <code>UART_STOPBITS_1_5</code> | UART frame with 1.5 stop bits |
| <code>UART_STOPBITS_2</code>   | UART frame with 2 stop bits   |

## **UART Advanced Feature Stop Mode Enable**

|                                  |                        |
|----------------------------------|------------------------|
| UART_ADVFEATURE_STOPMODE_DISABLE | UART stop mode disable |
| UART_ADVFEATURE_STOPMODE_ENABLE  | UART stop mode enable  |

### **UART polling-based communications time-out value**

**HAL\_UART\_TIMEOUT\_VALUE**    UART polling-based communications time-out value

## **UART Advanced Feature TX Pin Active Level Inversion**

|                               |                                       |
|-------------------------------|---------------------------------------|
| UART_ADVFEATURE_TXINV_DISABLE | TX pin active level inversion disable |
| UART_ADVFEATURE_TXINV_ENABLE  | TX pin active level inversion enable  |

### **UART WakeUp Address Length**

UART\_ADDRESS\_DETECT\_4B 4-bit long wake-up address  
UART\_ADDRESS\_DETECT\_7B 7-bit long wake-up address

## **UART WakeUp From Stop Selection**

|                                  |                                                 |
|----------------------------------|-------------------------------------------------|
| UART_WAKEUP_ON_ADDRESS           | UART wake-up on address                         |
| UART_WAKEUP_ON_STARTBIT          | UART wake-up on start bit                       |
| UART_WAKEUP_ON_READDATA_NONEMPTY | UART wake-up on receive data register not empty |

## **UART WakeUp Methods**

|                               |                              |
|-------------------------------|------------------------------|
| UART_WAKEUPMETHOD_IDLELINE    | UART wake-up on idle line    |
| UART_WAKEUPMETHOD_ADDRESSMARK | UART wake-up on address mark |

## 55 HAL UART Extension Driver

### 55.1 UARTEEx Firmware driver API description

#### 55.1.1 UART peripheral extended features

#### 55.1.2 Initialization and Configuration functions

This subsection provides a set of functions allowing to initialize the USARTx or the UARTy in asynchronous mode.

- For the asynchronous mode the parameters below can be configured:
  - Baud Rate
  - Word Length (Fixed to 8-bits only for LIN mode)
  - Stop Bit
  - Parity
  - Hardware flow control
  - Receiver/transmitter modes
  - Over Sampling Method
  - One-Bit Sampling Method
- For the asynchronous mode, the following advanced features can be configured as well:
  - TX and/or RX pin level inversion
  - data logical level inversion
  - RX and TX pins swap
  - RX overrun detection disabling
  - DMA disabling on RX error
  - MSB first on communication line
  - auto Baud rate detection

The HAL\_RS485Ex\_Init() API follows respectively the UART RS485 mode configuration procedures (details for the procedures are available in reference manual).

This section contains the following APIs:

- [\*\*HAL\\_RS485Ex\\_Init\(\)\*\*](#)

#### 55.1.3 IO operation function

This subsection provides functions allowing to manage the UART interrupts and to handle Wake up interrupt call-back.

1. Callback provided in No\_Blocking mode:
  - HAL\_UARTEEx\_WakeupCallback()

This section contains the following APIs:

- [\*\*HAL\\_UARTEEx\\_WakeupCallback\(\)\*\*](#)

#### 55.1.4 Peripheral Control functions

This subsection provides extended functions allowing to control the UART.

- HAL\_UARTEEx\_StopModeWakeUpSourceConfig() API sets Wakeup from Stop mode interrupt flag selection

- `HAL_UARTEx_EnableStopMode()` API allows the UART to wake up the MCU from Stop mode as long as UART clock is HSI or LSE
- `HAL_UARTEx_DisableStopMode()` API disables the above feature
- `HAL_MultiProcessorEx_AddressLength_Set()` API optionally sets the UART node address detection length to more than 4 bits for multiprocessor address mark wake up.

This section contains the following APIs:

- `HAL_UARTEx_StopModeWakeUpSourceConfig()`
- `HAL_UARTEx_EnableStopMode()`
- `HAL_UARTEx_DisableStopMode()`
- `HAL_MultiProcessorEx_AddressLength_Set()`
- `HAL_UARTEx_WakeupCallback()`

### 55.1.5 Detailed description of functions

#### `HAL_RS485Ex_Init`

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_RS485Ex_Init(UART_HandleTypeDef * huart, uint32_t Polarity, uint32_t AssertionTime, uint32_t DeassertionTime)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Initialize the RS485 Driver enable feature according to the specified parameters in the <code>UART_InitTypeDef</code> and creates the associated handle.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>Polarity:</b> select the driver enable polarity. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>UART_DE_POLARITY_HIGH</code> DE signal is active high</li> <li>– <code>UART_DE_POLARITY_LOW</code> DE signal is active low</li> </ul> </li> <li>• <b>AssertionTime:</b> Driver Enable assertion time: 5-bit value defining the time between the activation of the DE (Driver Enable) signal and the beginning of the start bit. It is expressed in sample time units (1/8 or 1/16 bit time, depending on the oversampling rate)</li> <li>• <b>DeassertionTime:</b> Driver Enable deassertion time: 5-bit value defining the time between the end of the last stop bit, in a transmitted message, and the de-activation of the DE (Driver Enable) signal. It is expressed in sample time units (1/8 or 1/16 bit time, depending on the oversampling rate).</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

#### `HAL_UARTEx_StopModeWakeUpSourceConfig`

|                      |                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>HAL_StatusTypeDef HAL_UARTEx_StopModeWakeUpSourceConfig(UART_HandleTypeDef * huart, UART_WakeUpTypeDef WakeUpSelection)</code>                                                                                        |
| Function description | Set Wakeup from Stop mode interrupt flag selection.                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>WakeUpSelection:</b> address match, Start Bit detection or RXNE bit status. This parameter can be one of the following values:</li> </ul> |

- UART\_WAKEUP\_ON\_ADDRESS
- UART\_WAKEUP\_ON\_STARTBIT
- UART\_WAKEUP\_ON\_READDATA\_NONEMPTY

**Return values**

- **HAL:** status

### **HAL\_UARTEx\_EnableStopMode**

|                      |                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UARTEx_EnableStopMode<br/>(UART_HandleTypeDef * huart)</b>                                                         |
| Function description | Enable UART Stop Mode.                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>                                                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• The UART is able to wake up the MCU from Stop mode as long as UART clock is HSI or LSE.</li> </ul> |

### **HAL\_UARTEx\_DisableStopMode**

|                      |                                                                                      |
|----------------------|--------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_UARTEx_DisableStopMode<br/>(UART_HandleTypeDef * huart)</b> |
| Function description | Disable UART Stop Mode.                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> </ul>       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>               |

### **HAL\_MultiProcessorEx\_AddressLength\_Set**

|                      |                                                                                                                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef<br/>HAL_MultiProcessorEx_AddressLength_Set<br/>(UART_HandleTypeDef * huart, uint32_t AddressLength)</b>                                                                                                                                                                                                     |
| Function description | By default in multiprocessor mode, when the wake up method is set to address mark, the UART handles only 4-bit long addresses detection; this API allows to enable longer addresses detection (6-, 7- or 8-bit long).                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>huart:</b> UART handle.</li> <li>• <b>AddressLength:</b> this parameter can be one of the following values: <ul style="list-style-type: none"> <li>- UART_ADDRESS_DETECT_4B 4-bit long address</li> <li>- UART_ADDRESS_DETECT_7B 6-, 7- or 8-bit long address</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• Addresses detection lengths are: 6-bit address detection in 7-bit data mode, 7-bit address detection in 8-bit data mode, 8-bit address detection in 9-bit data mode.</li> </ul>                                                                                                         |

**HAL\_UARTEx\_WakeupCallback**

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <b>void HAL_UARTEx_WakeupCallback (UART_HandleTypeDef *<br/>huart)</b>       |
| Function description | UART wakeup from Stop mode callback.                                         |
| Parameters           | <ul style="list-style-type: none"><li>• <b>huart:</b> UART handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                |

## 55.2 UARTE Firmware driver defines

### 55.2.1 UARTE

*UARTE Word Length*

UART\_WORDLENGTH\_8B 8-bit long UART frame

UART\_WORDLENGTH\_9B 9-bit long UART frame

## 56 HAL USART Generic Driver

### 56.1 USART Firmware driver registers structures

#### 56.1.1 USART\_InitTypeDef

##### Data Fields

- *uint32\_t BaudRate*
- *uint32\_t WordLength*
- *uint32\_t StopBits*
- *uint32\_t Parity*
- *uint32\_t Mode*
- *uint32\_t CLKPolarity*
- *uint32\_t CLKPhase*
- *uint32\_t CLKLastBit*

##### Field Documentation

- ***uint32\_t USART\_InitTypeDef::BaudRate***

This member configures the Usart communication baud rate. The baud rate is computed using the following formula: Baud Rate Register = ((PCLKx) / ((huart->Init.BaudRate))).

- ***uint32\_t USART\_InitTypeDef::WordLength***

Specifies the number of data bits transmitted or received in a frame. This parameter can be a value of [USARTEx\\_Word\\_Length](#).

- ***uint32\_t USART\_InitTypeDef::StopBits***

Specifies the number of stop bits transmitted. This parameter can be a value of [USART\\_Stop\\_Bits](#).

- ***uint32\_t USART\_InitTypeDef::Parity***

Specifies the parity mode. This parameter can be a value of [USART\\_Parity](#)  
**Note:**When parity is enabled, the computed parity is inserted at the MSB position of the transmitted data (9th bit when the word length is set to 9 data bits; 8th bit when the word length is set to 8 data bits).

- ***uint32\_t USART\_InitTypeDef::Mode***

Specifies whether the Receive or Transmit mode is enabled or disabled. This parameter can be a value of [USART\\_Mode](#).

- ***uint32\_t USART\_InitTypeDef::CLKPolarity***

Specifies the steady state of the serial clock. This parameter can be a value of [USART\\_Clock\\_Polarity](#).

- ***uint32\_t USART\_InitTypeDef::CLKPhase***

Specifies the clock transition on which the bit capture is made. This parameter can be a value of [USART\\_Clock\\_Phase](#).

- ***uint32\_t USART\_InitTypeDef::CLKLastBit***

Specifies whether the clock pulse corresponding to the last transmitted data bit (MSB) has to be output on the SCLK pin in synchronous mode. This parameter can be a value of [USART\\_Last\\_Bit](#).

#### 56.1.2 USART\_HandleTypeDef

##### Data Fields

- ***USART\_TypeDef \* Instance***
- ***USART\_InitTypeDef Init***

- *uint8\_t \* pTxBuffPtr*
- *uint16\_t TxXferSize*
- *\_IO uint16\_t TxXferCount*
- *uint8\_t \* pRxBuffPtr*
- *uint16\_t RxXferSize*
- *\_IO uint16\_t RxXferCount*
- *uint16\_t Mask*
- *DMA\_HandleTypeDef \* hdmatx*
- *DMA\_HandleTypeDef \* hdmarx*
- *HAL\_LockTypeDef Lock*
- *\_IO HAL\_USART\_StateTypeDef State*
- *\_IO uint32\_t ErrorCode*

#### Field Documentation

- ***USART\_TypeDef\* USART\_HandleTypeDef::Instance***  
USART registers base address
- ***USART\_InitTypeDef USART\_HandleTypeDef::Init***  
USART communication parameters
- ***uint8\_t\* USART\_HandleTypeDef::pTxBuffPtr***  
Pointer to USART Tx transfer Buffer
- ***uint16\_t USART\_HandleTypeDef::TxXferSize***  
USART Tx Transfer size
- ***\_IO uint16\_t USART\_HandleTypeDef::TxXferCount***  
USART Tx Transfer Counter
- ***uint8\_t\* USART\_HandleTypeDef::pRxBuffPtr***  
Pointer to USART Rx transfer Buffer
- ***uint16\_t USART\_HandleTypeDef::RxXferSize***  
USART Rx Transfer size
- ***\_IO uint16\_t USART\_HandleTypeDef::RxXferCount***  
USART Rx Transfer Counter
- ***uint16\_t USART\_HandleTypeDef::Mask***  
USART Rx RDR register mask
- ***DMA\_HandleTypeDef\* USART\_HandleTypeDef::hdmatx***  
USART Tx DMA Handle parameters
- ***DMA\_HandleTypeDef\* USART\_HandleTypeDef::hdmarx***  
USART Rx DMA Handle parameters
- ***HAL\_LockTypeDef USART\_HandleTypeDef::Lock***  
Locking object
- ***\_IO HAL\_USART\_StateTypeDef USART\_HandleTypeDef::State***  
USART communication state
- ***\_IO uint32\_t USART\_HandleTypeDef::ErrorCode***  
USART Error code

## 56.2 USART Firmware driver API description

### 56.2.1 How to use this driver

The USART HAL driver can be used as follows:

1. Declare a USART\_HandleTypeDef handle structure (eg. USART\_HandleTypeDef husart).
2. Initialize the USART low level resources by implementing the HAL\_USART\_MspInit() API:
  - Enable the USARTx interface clock.

- USART pins configuration:
    - Enable the clock for the USART GPIOs.
    - Configure these USART pins as alternate function pull-up.
  - NVIC configuration if you need to use interrupt process (HAL\_USART\_Transmit\_IT(), HAL\_USART\_Receive\_IT() and HAL\_USART\_TransmitReceive\_IT() APIs):
    - Configure the USARTx interrupt priority.
    - Enable the NVIC USART IRQ handle.
  - USART interrupts handling: The specific USART interrupts (Transmission complete interrupt, RXNE interrupt and Error Interrupts) will be managed using the macros \_\_HAL\_USART\_ENABLE\_IT() and \_\_HAL\_USART\_DISABLE\_IT() inside the transmit and receive process.
  - DMA Configuration if you need to use DMA process (HAL\_USART\_Transmit\_DMA() HAL\_USART\_Receive\_DMA() and HAL\_USART\_TransmitReceive\_DMA() APIs):
    - Declare a DMA handle structure for the Tx/Rx channel.
    - Enable the DMAx interface clock.
    - Configure the declared DMA handle structure with the required Tx/Rx parameters.
    - Configure the DMA Tx/Rx channel.
    - Associate the initialized DMA handle to the USART DMA Tx/Rx handle.
    - Configure the priority and enable the NVIC for the transfer complete interrupt on the DMA Tx/Rx channel.
3. Program the Baud Rate, Word Length, Stop Bit, Parity, Hardware flow control and Mode (Receiver/Transmitter) in the husart handle Init structure.
  4. Initialize the USART registers by calling the HAL\_USART\_Init() API:
    - This API configures also the low level Hardware (GPIO, CLOCK, CORTEX...etc) by calling the customized HAL\_USART\_MspInit(&husart) API.
  5. Three operation modes are available within this driver :

### **Polling mode IO operation**

- Send an amount of data in blocking mode using HAL\_USART\_Transmit()
- Receive an amount of data in blocking mode using HAL\_USART\_Receive()

### **Interrupt mode IO operation**

- Send an amount of data in non blocking mode using HAL\_USART\_Transmit\_IT()
- At transmission end of half transfer HAL\_USART\_TxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_USART\_TxHalfCpltCallback
- At transmission end of transfer HAL\_USART\_TxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_USART\_TxCpltCallback
- Receive an amount of data in non blocking mode using HAL\_USART\_Receive\_IT()
- At reception end of half transfer HAL\_USART\_RxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_USART\_RxHalfCpltCallback
- At reception end of transfer HAL\_USART\_RxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_USART\_RxCpltCallback
- In case of transfer Error, HAL\_USART\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_USART\_ErrorCallback

### DMA mode IO operation

- Send an amount of data in non blocking mode (DMA) using HAL\_USART\_Transmit\_DMA()
- At transmission end of half transfer HAL\_USART\_TxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_USART\_TxHalfCpltCallback
- At transmission end of transfer HAL\_USART\_TxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_USART\_TxCpltCallback
- Receive an amount of data in non blocking mode (DMA) using HAL\_USART\_Receive\_DMA()
- At reception end of half transfer HAL\_USART\_RxHalfCpltCallback is executed and user can add his own code by customization of function pointer HAL\_USART\_RxHalfCpltCallback
- At reception end of transfer HAL\_USART\_RxCpltCallback is executed and user can add his own code by customization of function pointer HAL\_USART\_RxCpltCallback
- In case of transfer Error, HAL\_USART\_ErrorCallback() function is executed and user can add his own code by customization of function pointer HAL\_USART\_ErrorCallback
- Pause the DMA Transfer using HAL\_USART\_DMAPause()
- Resume the DMA Transfer using HAL\_USART\_DMAResume()
- Stop the DMA Transfer using HAL\_USART\_DMAStop()

### USART HAL driver macros list

Below the list of most used macros in USART HAL driver.

- \_\_HAL\_USART\_ENABLE: Enable the USART peripheral
- \_\_HAL\_USART\_DISABLE: Disable the USART peripheral
- \_\_HAL\_USART\_GET\_FLAG : Check whether the specified USART flag is set or not
- \_\_HAL\_USART\_CLEAR\_FLAG : Clear the specified USART pending flag
- \_\_HAL\_USART\_ENABLE\_IT: Enable the specified USART interrupt
- \_\_HAL\_USART\_DISABLE\_IT: Disable the specified USART interrupt



You can refer to the USART HAL driver header file for more useful macros



To configure and enable/disable the USART to wake up the MCU from stop mode, resort to UART API's HAL\_UARTEStopModeWakeUpSourceConfig(), HAL\_UARTEEnableStopMode() and HAL\_UARTEDisableStopMode() in casting the USART handle to UART type UART\_HandleTypeDef.

#### 56.2.2 Initialization and Configuration functions

This subsection provides a set of functions allowing to initialize the USART in asynchronous and in synchronous modes.

- For the asynchronous mode only these parameters can be configured:
  - Baud Rate
  - Word Length
  - Stop Bit

- Parity
- USART polarity
- USART phase
- USART LastBit
- Receiver/transmitter modes

The HAL\_USART\_Init() function follows the USART synchronous configuration procedure (details for the procedure are available in reference manual).

This section contains the following APIs:

- [\*\*HAL\\_USART\\_Init\(\)\*\*](#)
- [\*\*HAL\\_USART\\_DelInit\(\)\*\*](#)
- [\*\*HAL\\_USART\\_MspInit\(\)\*\*](#)
- [\*\*HAL\\_USART\\_MspDelInit\(\)\*\*](#)

### 56.2.3 IO operation functions

This subsection provides a set of functions allowing to manage the USART synchronous data transfers.

The USART supports master mode only: it cannot receive or send data related to an input clock (SCLK is always an output).

1. There are two modes of transfer:
  - Blocking mode: The communication is performed in polling mode. The HAL status of all data processing is returned by the same function after finishing transfer.
  - No-Blocking mode: The communication is performed using Interrupts or DMA, These APIs return the HAL status. The end of the data processing will be indicated through the dedicated USART IRQ when using Interrupt mode or the DMA IRQ when using DMA mode. The HAL\_USART\_TxCpltCallback(), HAL\_USART\_RxCpltCallback() and HAL\_USART\_TxRxCpltCallback() user callbacks will be executed respectively at the end of the transmit or Receive process The HAL\_USART\_ErrorCallback() user callback will be executed when a communication error is detected
2. Blocking mode APIs are :
  - HAL\_USART\_Transmit() in simplex mode
  - HAL\_USART\_Receive() in full duplex receive only
  - HAL\_USART\_TransmitReceive() in full duplex mode
3. No-Blocking mode APIs with Interrupt are :
  - HAL\_USART\_Transmit\_IT() in simplex mode
  - HAL\_USART\_Receive\_IT() in full duplex receive only
  - HAL\_USART\_TransmitReceive\_IT() in full duplex mode
  - HAL\_USART\_IRQHandler()
4. No-Blocking mode APIs with DMA are :
  - HAL\_USART\_Transmit\_DMA() in simplex mode
  - HAL\_USART\_Receive\_DMA() in full duplex receive only
  - HAL\_USART\_TransmitReceive\_DMA() in full duplex mode
  - HAL\_USART\_DMAPause()
  - HAL\_USART\_DMAResume()
  - HAL\_USART\_DMAStop()
5. A set of Transfer Complete Callbacks are provided in No-Blocking mode:
  - HAL\_USART\_TxCpltCallback()
  - HAL\_USART\_RxCpltCallback()
  - HAL\_USART\_TxHalfCpltCallback()
  - HAL\_USART\_RxHalfCpltCallback()

- HAL\_USART\_ErrorCallback()
  - HAL\_USART\_TxRxCpltCallback()
6. Non-Blocking mode transfers could be aborted using Abort API's :
- HAL\_USART\_Abort()
  - HAL\_USART\_Abort\_IT()
7. For Abort services based on interrupts (HAL\_USART\_Abort\_IT), a Abort Complete Callbacks is provided:
- HAL\_USART\_AbortCpltCallback()
8. In Non-Blocking mode transfers, possible errors are split into 2 categories. Errors are handled as follows :
- Error is considered as Recoverable and non blocking : Transfer could go till end, but error severity is to be evaluated by user : this concerns Frame Error, Parity Error or Noise Error in Interrupt mode reception . Received character is then retrieved and stored in Rx buffer, Error code is set to allow user to identify error type, and HAL\_USART\_ErrorCallback() user callback is executed. Transfer is kept ongoing on USART side. If user wants to abort it, Abort services should be called by user.
  - Error is considered as Blocking : Transfer could not be completed properly and is aborted. This concerns Overrun Error In Interrupt mode reception and all errors in DMA mode. Error code is set to allow user to identify error type, and HAL\_USART\_ErrorCallback() user callback is executed.

This section contains the following APIs:

- [`HAL\_USART\_Transmit\(\)`](#)
- [`HAL\_USART\_Receive\(\)`](#)
- [`HAL\_USART\_TransmitReceive\(\)`](#)
- [`HAL\_USART\_Transmit\_IT\(\)`](#)
- [`HAL\_USART\_Receive\_IT\(\)`](#)
- [`HAL\_USART\_TransmitReceive\_IT\(\)`](#)
- [`HAL\_USART\_Transmit\_DMA\(\)`](#)
- [`HAL\_USART\_Receive\_DMA\(\)`](#)
- [`HAL\_USART\_TransmitReceive\_DMA\(\)`](#)
- [`HAL\_USART\_DMAPause\(\)`](#)
- [`HAL\_USART\_DMAResume\(\)`](#)
- [`HAL\_USART\_DMAStop\(\)`](#)
- [`HAL\_USART\_Abort\(\)`](#)
- [`HAL\_USART\_Abort\_IT\(\)`](#)
- [`HAL\_USART\_IRQHandler\(\)`](#)
- [`HAL\_USART\_TxCpltCallback\(\)`](#)
- [`HAL\_USART\_TxHalfCpltCallback\(\)`](#)
- [`HAL\_USART\_RxCpltCallback\(\)`](#)
- [`HAL\_USART\_RxHalfCpltCallback\(\)`](#)
- [`HAL\_USART\_TxRxCpltCallback\(\)`](#)
- [`HAL\_USART\_ErrorCallback\(\)`](#)
- [`HAL\_USART\_AbortCpltCallback\(\)`](#)

#### 56.2.4 Peripheral State and Error functions

This subsection provides functions allowing to :

- Return the USART handle state
- Return the USART handle error code

This section contains the following APIs:

- [\*\*\*HAL\\_USART\\_GetState\(\)\*\*\*](#)
- [\*\*\*HAL\\_USART\\_GetError\(\)\*\*\*](#)

### 56.2.5 Detailed description of functions

#### **HAL\_USART\_Init**

|                      |                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_Init<br/>(USART_HandleTypeDef * huart)</b>                                                      |
| Function description | Initialize the USART mode according to the specified parameters in the USART_InitTypeDef and initialize the associated handle. |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> </ul>                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                         |

#### **HAL\_USART\_DelInit**

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_DelInit<br/>(USART_HandleTypeDef * huart)</b>     |
| Function description | Deinitialize the USART peripheral.                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>           |

#### **HAL\_USART\_MspInit**

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <b>void HAL_USART_MspInit (USART_HandleTypeDef * huart)</b>                      |
| Function description | Initialize the USART MSP.                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                  |

#### **HAL\_USART\_MspDelInit**

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <b>void HAL_USART_MspDelInit (USART_HandleTypeDef * huart)</b>                   |
| Function description | Deinitialize the USART MSP.                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                  |

#### **HAL\_USART\_Transmit**

|                      |                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_Transmit<br/>(USART_HandleTypeDef * huart, uint8_t * pTxData, uint16_t<br/>Size, uint32_t Timeout)</b> |
| Function description | Simplex send an amount of data in blocking mode.                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> <li>• <b>pTxData:</b> Pointer to data buffer.</li> </ul>   |

- **Size:** Amount of data to be sent.
  - **Timeout:** Timeout duration.
- Return values
- **HAL:** status

### **HAL\_USART\_Receive**

|                      |                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_Receive</b><br><b>(USART_HandleTypeDef * husart, uint8_t * pRxData, uint16_t Size, uint32_t Timeout)</b>                                                                                                 |
| Function description | Receive an amount of data in blocking mode.                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> <li>• <b>pRxData:</b> Pointer to data buffer.</li> <li>• <b>Size:</b> Amount of data to be received.</li> <li>• <b>Timeout:</b> Timeout duration.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                    |
| Notes                | • To receive synchronous data, dummy data are simultaneously transmitted.                                                                                                                                                               |

### **HAL\_USART\_TransmitReceive**

|                      |                                                                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_TransmitReceive</b><br><b>(USART_HandleTypeDef * husart, uint8_t * pTxData, uint8_t * pRxData, uint16_t Size, uint32_t Timeout)</b>                                                                                                                                                        |
| Function description | Full-Duplex Send and Receive an amount of data in blocking mode.                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> <li>• <b>pTxData:</b> pointer to TX data buffer.</li> <li>• <b>pRxData:</b> pointer to RX data buffer.</li> <li>• <b>Size:</b> amount of data to be sent (same amount to be received).</li> <li>• <b>Timeout:</b> Timeout duration.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                                                                                                                                                      |

### **HAL\_USART\_Transmit\_IT**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_Transmit_IT</b><br><b>(USART_HandleTypeDef * husart, uint8_t * pTxData, uint16_t Size)</b>                                                              |
| Function description | Send an amount of data in interrupt mode.                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> <li>• <b>pTxData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be sent.</li> </ul> |
| Return values        | • <b>HAL:</b> status                                                                                                                                                                   |

**HAL\_USART\_Receive\_IT**

|                      |                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_Receive_IT<br/>(USART_HandleTypeDef * husart, uint8_t * pRxData, uint16_t Size)</b>                                                                         |
| Function description | Receive an amount of data in interrupt mode.                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> <li>• <b>pRxData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be received.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• To receive synchronous data, dummy data are simultaneously transmitted.</li> </ul>                                                                |

**HAL\_USART\_TransmitReceive\_IT**

|                      |                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_TransmitReceive_IT<br/>(USART_HandleTypeDef * husart, uint8_t * pTxData, uint8_t * pRxData, uint16_t Size)</b>                                                                                                                                |
| Function description | Full-Duplex Send and Receive an amount of data in interrupt mode.                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> <li>• <b>pTxData:</b> pointer to TX data buffer.</li> <li>• <b>pRxData:</b> pointer to RX data buffer.</li> <li>• <b>Size:</b> amount of data to be sent (same amount to be received).</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                                                                       |

**HAL\_USART\_Transmit\_DMA**

|                      |                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_Transmit_DMA<br/>(USART_HandleTypeDef * husart, uint8_t * pTxData, uint16_t Size)</b>                                                                                                   |
| Function description | Send an amount of data in DMA mode.                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>husart:</b> USART handle.</li> <li>• <b>pTxData:</b> pointer to data buffer.</li> <li>• <b>Size:</b> amount of data to be sent.</li> </ul>                                 |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                                                 |
| Notes                | <ul style="list-style-type: none"> <li>• This function starts a DMA transfer in interrupt mode meaning that DMA half transfer complete, DMA transfer complete and DMA transfer error interrupts are enabled</li> </ul> |

**HAL\_USART\_Receive\_DMA**

|                      |                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_Receive_DMA<br/>(USART_HandleTypeDef * husart, uint8_t * pRxData, uint16_t Size)</b> |
| Function description | Receive an amount of data in DMA mode.                                                                              |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li><b>husart:</b> USART handle.</li> <li><b>pRxData:</b> pointer to data buffer.</li> <li><b>Size:</b> amount of data to be received.</li> </ul>                                                                                                                                                                                                                                                        |
| Return values | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                                                                                                                                        |
| Notes         | <ul style="list-style-type: none"> <li>When the USART parity is enabled (PCE = 1), the received data contain the parity bit (MSB position).</li> <li>The USART DMA transmit channel must be configured in order to generate the clock for the slave.</li> <li>This function starts a DMA transfer in interrupt mode meaning that DMA half transfer complete, DMA transfer complete and DMA transfer error interrupts are enabled</li> </ul> |

### HAL\_USART\_TransmitReceive\_DMA

|                      |                                                                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_TransmitReceive_DMA(USART_HandleTypeDef * husart, uint8_t * pTxData, uint8_t * pRxData, uint16_t Size)</b>                                                                                                                                                                             |
| Function description | Full-Duplex Transmit Receive an amount of data in non-blocking mode.                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>husart:</b> USART handle.</li> <li><b>pTxData:</b> pointer to TX data buffer.</li> <li><b>pRxData:</b> pointer to RX data buffer.</li> <li><b>Size:</b> amount of data to be received/sent.</li> </ul>                                                                      |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>                                                                                                                                                                                                                                                  |
| Notes                | <ul style="list-style-type: none"> <li>When the USART parity is enabled (PCE = 1) the data received contain the parity bit.</li> <li>This function starts a 2 DMA transfers in interrupt mode meaning that DMA half transfer complete, DMA transfer complete and DMA transfer error interrupts are enabled</li> </ul> |

### HAL\_USART\_DMAPause

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_DMAPause(USART_HandleTypeDef * husart)</b>      |
| Function description | Pause the DMA Transfer.                                                        |
| Parameters           | <ul style="list-style-type: none"> <li><b>husart:</b> USART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>           |

### HAL\_USART\_DMAResume

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_USART_DMAResume(USART_HandleTypeDef * husart)</b>     |
| Function description | Resume the DMA Transfer.                                                       |
| Parameters           | <ul style="list-style-type: none"> <li><b>husart:</b> USART handle.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>HAL:</b> status</li> </ul>           |

**HAL\_USART\_DMAMain**

Function name      **HAL\_StatusTypeDef HAL\_USART\_DMAMain  
(USART\_HandleTypeDef \* huart)**

Function description      Stop the DMA Transfer.

Parameters      •    **huart:** USART handle.

Return values      •    **HAL:** status

**HAL\_USART\_Abort**

Function name      **HAL\_StatusTypeDef HAL\_USART\_Abort  
(USART\_HandleTypeDef \* huart)**

Function description      Abort ongoing transfers (blocking mode).

Parameters      •    **huart:** USART handle.

Return values      •    **HAL:** status

Notes      •    This procedure could be used for aborting any ongoing transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable USART Interrupts (Tx and Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL\_DMA\_Abort (in case of transfer in DMA mode)Set handle State to READY  
•    This procedure is executed in blocking mode : when exiting function, Abort is considered as completed.

**HAL\_USART\_Abort\_IT**

Function name      **HAL\_StatusTypeDef HAL\_USART\_Abort\_IT  
(USART\_HandleTypeDef \* huart)**

Function description      Abort ongoing transfers (Interrupt mode).

Parameters      •    **huart:** USART handle.

Return values      •    **HAL:** status

Notes      •    This procedure could be used for aborting any ongoing transfer started in Interrupt or DMA mode. This procedure performs following operations : Disable USART Interrupts (Tx and Rx)Disable the DMA transfer in the peripheral register (if enabled)Abort DMA transfer by calling HAL\_DMA\_Abort\_IT (in case of transfer in DMA mode)Set handle State to READYAt abort completion, call user abort complete callback  
•    This procedure is executed in Interrupt mode, meaning that abort procedure could be considered as completed only when user abort complete callback is executed (not when exiting function).

**HAL\_USART\_IRQHandler**

Function name      **void HAL\_USART\_IRQHandler (USART\_HandleTypeDef \*  
                        huart)**

Function description      Handle USART interrupt request.

---

|               |                                                                                |
|---------------|--------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"><li>• <b>husart:</b> USART handle.</li></ul> |
| Return values | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                  |

### **HAL\_USART\_TxCpltCallback**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>void HAL_USART_TxCpltCallback (USART_HandleTypeDef * husart)</b>            |
| Function description | Tx Transfer completed callback.                                                |
| Parameters           | <ul style="list-style-type: none"><li>• <b>husart:</b> USART handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                  |

### **HAL\_USART\_TxHalfCpltCallback**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>void HAL_USART_TxHalfCpltCallback (USART_HandleTypeDef * husart)</b>        |
| Function description | Tx Half Transfer completed callback.                                           |
| Parameters           | <ul style="list-style-type: none"><li>• <b>husart:</b> USART handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                  |

### **HAL\_USART\_RxCpltCallback**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>void HAL_USART_RxCpltCallback (USART_HandleTypeDef * husart)</b>            |
| Function description | Rx Transfer completed callback.                                                |
| Parameters           | <ul style="list-style-type: none"><li>• <b>husart:</b> USART handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                  |

### **HAL\_USART\_RxHalfCpltCallback**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>void HAL_USART_RxHalfCpltCallback (USART_HandleTypeDef * husart)</b>        |
| Function description | Rx Half Transfer completed callback.                                           |
| Parameters           | <ul style="list-style-type: none"><li>• <b>husart:</b> USART handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                  |

### **HAL\_USART\_TxRxCpltCallback**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <b>void HAL_USART_TxRxCpltCallback (USART_HandleTypeDef * husart)</b>          |
| Function description | Tx/Rx Transfers completed callback for the non-blocking process.               |
| Parameters           | <ul style="list-style-type: none"><li>• <b>husart:</b> USART handle.</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                  |

**HAL\_USART\_ErrorCallback**

Function name      **void HAL\_USART\_ErrorCallback (USART\_HandleTypeDef \*  
                      husart)**

Function description      USART error callback.

Parameters      •    **husart:** USART handle.

Return values      •    **None**

**HAL\_USART\_AbortCpltCallback**

Function name      **void HAL\_USART\_AbortCpltCallback (USART\_HandleTypeDef  
                      \* husart)**

Function description      USART Abort Complete callback.

Parameters      •    **husart:** USART handle.

Return values      •    **None**

**HAL\_USART\_GetState**

Function name      **HAL\_USART\_StateTypeDef HAL\_USART\_GetState  
(USART\_HandleTypeDef \* husart)**

Function description      Return the USART handle state.

Parameters      •    **husart:** pointer to a USART\_HandleTypeDef structure that  
                      contains the configuration information for the specified  
                      USART.

Return values      •    **USART:** handle state

**HAL\_USART\_GetError**

Function name      **uint32\_t HAL\_USART\_GetError (USART\_HandleTypeDef \*  
                      husart)**

Function description      Return the USART error code.

Parameters      •    **husart:** pointer to a USART\_HandleTypeDef structure that  
                      contains the configuration information for the specified  
                      USART.

Return values      •    **USART:** handle Error Code

## 56.3 USART Firmware driver defines

### 56.3.1 USART

#### *USART Clock*

**USART\_CLOCK\_DISABLE**    USART clock disable

**USART\_CLOCK\_ENABLE**    USART clock enable

#### *USART Clock Phase*

**USART\_PHASE\_1EDGE**    USART frame phase on first clock transition

**USART\_PHASE\_2EDGE** USART frame phase on second clock transition

#### **USART Clock Polarity**

**USART\_POLARITY\_LOW** USART Clock signal is steady Low

**USART\_POLARITY\_HIGH** USART Clock signal is steady High

#### **USART Error**

**HAL\_USART\_ERROR\_NONE** No error

**HAL\_USART\_ERROR\_PE** Parity error

**HAL\_USART\_ERROR\_NE** Noise error

**HAL\_USART\_ERROR\_FE** frame error

**HAL\_USART\_ERROR\_ORE** Overrun error

**HAL\_USART\_ERROR\_DMA** DMA transfer error

#### **USART Exported Macros**

| <b>__HAL_USART_RESET_HANDLE_STATE</b> | <b>Description:</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                       | <ul style="list-style-type: none"> <li>• Reset USART handle state.</li> </ul> <b>Parameters:</b> <ul style="list-style-type: none"> <li>• <b>__HANDLE__</b>: USART handle.</li> </ul> <b>Return value:</b> <ul style="list-style-type: none"> <li>• None</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>__HAL_USART_FLUSH_DRREGISTER</b>   | <b>Description:</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|                                       | <ul style="list-style-type: none"> <li>• Flush the USART Data registers.</li> </ul> <b>Parameters:</b> <ul style="list-style-type: none"> <li>• <b>__HANDLE__</b>: specifies the USART Handle.</li> </ul> <b>Return value:</b> <ul style="list-style-type: none"> <li>• None</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>__HAL_USART_GET_FLAG</b>           | <b>Description:</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|                                       | <ul style="list-style-type: none"> <li>• Check whether the specified USART flag is set or not.</li> </ul> <b>Parameters:</b> <ul style="list-style-type: none"> <li>• <b>__HANDLE__</b>: specifies the USART Handle</li> <li>• <b>__FLAG__</b>: specifies the flag to check. This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– USART_FLAG_RXEACK Receive enable acknowledge flag</li> <li>– USART_FLAG_TEACK Transmit enable acknowledge flag</li> <li>– USART_FLAG_BUSY Busy flag</li> <li>– USART_FLAG_CTS CTS Change flag</li> <li>– USART_FLAG_TXE Transmit data register empty flag</li> <li>– USART_FLAG_TC Transmission Complete flag</li> </ul> </li> </ul> |

- USART\_FLAG\_RXNE Receive data register not empty flag
- USART\_FLAG\_IDLE Idle Line detection flag
- USART\_FLAG\_ORE OverRun Error flag
- USART\_FLAG\_NE Noise Error flag
- USART\_FLAG\_FE Framing Error flag
- USART\_FLAG\_PE Parity Error flag

**Return value:**

- The new state of \_\_FLAG\_\_ (TRUE or FALSE).

[\\_\\_HAL\\_USART\\_CLEAR\\_FLAG](#)**Description:**

- Clear the specified USART pending flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the USART Handle.
- \_\_FLAG\_\_: specifies the flag to check. This parameter can be any combination of the following values:
  - USART\_CLEAR\_PEF
  - USART\_CLEAR\_FEF
  - USART\_CLEAR\_NEF
  - USART\_CLEAR\_OREF
  - USART\_CLEAR\_IDLEF
  - USART\_CLEAR\_TCF
  - USART\_CLEAR\_CTSF

**Return value:**

- None

[\\_\\_HAL\\_USART\\_CLEAR\\_PEEFLAG](#)**Description:**

- Clear the USART PE pending flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the USART Handle.

**Return value:**

- None

[\\_\\_HAL\\_USART\\_CLEAR\\_FEFLAG](#)**Description:**

- Clear the USART FE pending flag.

**Parameters:**

- \_\_HANDLE\_\_: specifies the USART Handle.

**Return value:**

- None

[\\_\\_HAL\\_USART\\_CLEAR\\_NEFLAG](#)**Description:**

- Clear the USART NE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.

**Return value:**

- None

`__HAL_USART_CLEAR_OREFLAG`

**Description:**

- Clear the USART ORE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.

**Return value:**

- None

`__HAL_USART_CLEAR_IDLEFLAG`

**Description:**

- Clear the USART IDLE pending flag.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.

**Return value:**

- None

`__HAL_USART_ENABLE_IT`

**Description:**

- Enable the specified USART interrupt.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.
- `__INTERRUPT__`: specifies the USART interrupt source to enable. This parameter can be one of the following values:
  - USART\_IT\_TXE Transmit Data Register empty interrupt
  - USART\_IT\_TC Transmission complete interrupt
  - USART\_IT\_RXNE Receive Data register not empty interrupt
  - USART\_IT\_IDLE Idle line detection interrupt
  - USART\_IT\_PE Parity Error interrupt
  - USART\_IT\_ERR Error interrupt(Frame error, noise error, overrun error)

**Return value:**

- None

`__HAL_USART_DISABLE_IT`

**Description:**

- Disable the specified USART interrupt.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.
- `__INTERRUPT__`: specifies the USART

interrupt source to disable. This parameter can be one of the following values:

- USART\_IT\_TXE Transmit Data Register empty interrupt
- USART\_IT\_TC Transmission complete interrupt
- USART\_IT\_RXNE Receive Data register not empty interrupt
- USART\_IT\_IDLE Idle line detection interrupt
- USART\_IT\_PE Parity Error interrupt
- USART\_IT\_ERR Error interrupt(Frame error, noise error, overrun error)

#### **Return value:**

- None

### [\\_\\_HAL\\_USART\\_GET\\_IT](#)

#### **Description:**

- Check whether the specified USART interrupt has occurred or not.

#### **Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the USART Handle.
- [\\_\\_IT\\_\\_](#): specifies the USART interrupt source to check. This parameter can be one of the following values:
  - USART\_IT\_TXE Transmit Data Register empty interrupt
  - USART\_IT\_TC Transmission complete interrupt
  - USART\_IT\_RXNE Receive Data register not empty interrupt
  - USART\_IT\_IDLE Idle line detection interrupt
  - USART\_IT\_ORE OverRun Error interrupt
  - USART\_IT\_NE Noise Error interrupt
  - USART\_IT\_FE Framing Error interrupt
  - USART\_IT\_PE Parity Error interrupt

#### **Return value:**

- The: new state of [\\_\\_IT\\_\\_](#) (TRUE or FALSE).

### [\\_\\_HAL\\_USART\\_GET\\_IT\\_SOURCE](#)

#### **Description:**

- Check whether the specified USART interrupt source is enabled or not.

#### **Parameters:**

- [\\_\\_HANDLE\\_\\_](#): specifies the USART Handle.
- [\\_\\_IT\\_\\_](#): specifies the USART interrupt source to check. This parameter can be one of the following values:
  - USART\_IT\_TXE Transmit Data Register empty interrupt

- USART\_IT\_TC Transmission complete interrupt
- USART\_IT\_RXNE Receive Data register not empty interrupt
- USART\_IT\_IDLE Idle line detection interrupt
- USART\_IT\_ORE OverRun Error interrupt
- USART\_IT\_NE Noise Error interrupt
- USART\_IT\_FE Framing Error interrupt
- USART\_IT\_PE Parity Error interrupt

**Return value:**

- The new state of `__IT__` (TRUE or FALSE).

`__HAL_USART_CLEAR_IT`**Description:**

- Clear the specified USART ISR flag, in setting the proper ICR register flag.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.
- `__IT_CLEAR__`: specifies the interrupt clear register flag that needs to be set to clear the corresponding interrupt. This parameter can be one of the following values:
  - `USART_CLEAR_PEF` Parity Error Clear Flag
  - `USART_CLEAR_FEF` Framing Error Clear Flag
  - `USART_CLEAR_NEF` Noise detected Clear Flag
  - `USART_CLEAR_OREF` OverRun Error Clear Flag
  - `USART_CLEAR_IDLEF` IDLE line detected Clear Flag
  - `USART_CLEAR_TCF` Transmission Complete Clear Flag
  - `USART_CLEAR_CTSF` CTS Interrupt Clear Flag

**Return value:**

- None

`__HAL_USART_SEND_REQ`**Description:**

- Set a specific USART request flag.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.
- `__REQ__`: specifies the request flag to set. This parameter can be one of the following values:
  - `USART_RXDATA_FLUSH_REQUEST` Receive Data flush Request
  - `USART_TXDATA_FLUSH_REQUEST`

---

Transmit data flush Request**Return value:**

- None

`__HAL_USART_ONE_BIT_SAMPLE  
_ENABLE`

**Description:**

- Enable the USART one bit sample method.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.

**Return value:**

- None

`__HAL_USART_ONE_BIT_SAMPLE  
_DISABLE`

**Description:**

- Disable the USART one bit sample method.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.

**Return value:**

- None

`__HAL_USART_ENABLE`

**Description:**

- Enable USART.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.

**Return value:**

- None

`__HAL_USART_DISABLE`

**Description:**

- Disable USART.

**Parameters:**

- `__HANDLE__`: specifies the USART Handle.

**Return value:**

- None

**USART Flags**

|                               |                                        |
|-------------------------------|----------------------------------------|
| <code>USART_FLAG_RXACK</code> | USART receive enable acknowledge flag  |
| <code>USART_FLAG_TEACK</code> | USART transmit enable acknowledge flag |
| <code>USART_FLAG_BUSY</code>  | USART busy flag                        |
| <code>USART_FLAG_CTS</code>   | USART clear to send flag               |
| <code>USART_FLAG_CTSIF</code> | USART clear to send interrupt flag     |
| <code>USART_FLAG_LBDF</code>  | USART LIN break detection flag         |
| <code>USART_FLAG_TXE</code>   | USART transmit data register empty     |
| <code>USART_FLAG_TC</code>    | USART transmission complete            |

---

|                              |                                    |
|------------------------------|------------------------------------|
| <code>USART_FLAG_RXNE</code> | USART read data register not empty |
| <code>USART_FLAG_IDLE</code> | USART idle flag                    |
| <code>USART_FLAG_ORE</code>  | USART overrun error                |
| <code>USART_FLAG_NE</code>   | USART noise error                  |
| <code>USART_FLAG_FE</code>   | USART frame error                  |
| <code>USART_FLAG_PE</code>   | USART parity error                 |

***USART Interruption Flags Mask***

|                            |                                |
|----------------------------|--------------------------------|
| <code>USART_IT_MASK</code> | USART interruptions flags mask |
|----------------------------|--------------------------------|

***USART Interrupts Definition***

|                            |                                                 |
|----------------------------|-------------------------------------------------|
| <code>USART_IT_PE</code>   | USART parity error interruption                 |
| <code>USART_IT_TXE</code>  | USART transmit data register empty interruption |
| <code>USART_IT_TC</code>   | USART transmission complete interruption        |
| <code>USART_IT_RXNE</code> | USART read data register not empty interruption |
| <code>USART_IT_IDLE</code> | USART idle interruption                         |
| <code>USART_IT_ERR</code>  | USART error interruption                        |
| <code>USART_IT_ORE</code>  | USART overrun error interruption                |
| <code>USART_IT_NE</code>   | USART noise error interruption                  |
| <code>USART_IT_FE</code>   | USART frame error interruption                  |

***USART Interruption Clear Flags***

|                                |                                  |
|--------------------------------|----------------------------------|
| <code>USART_CLEAR_PEF</code>   | Parity Error Clear Flag          |
| <code>USART_CLEAR_FEF</code>   | Framing Error Clear Flag         |
| <code>USART_CLEAR_NEF</code>   | Noise detected Clear Flag        |
| <code>USART_CLEAR_OREF</code>  | OverRun Error Clear Flag         |
| <code>USART_CLEAR_IDLEF</code> | IDLE line detected Clear Flag    |
| <code>USART_CLEAR_TCF</code>   | Transmission Complete Clear Flag |
| <code>USART_CLEAR_CTSF</code>  | CTS Interrupt Clear Flag         |

***USART Last Bit***

|                                    |                                                              |
|------------------------------------|--------------------------------------------------------------|
| <code>USART_LASTBIT_DISABLE</code> | USART frame last data bit clock pulse not output to SCLK pin |
|------------------------------------|--------------------------------------------------------------|

|                                   |                                                          |
|-----------------------------------|----------------------------------------------------------|
| <code>USART_LASTBIT_ENABLE</code> | USART frame last data bit clock pulse output to SCLK pin |
|-----------------------------------|----------------------------------------------------------|

***USART Mode***

|                               |                |
|-------------------------------|----------------|
| <code>USART_MODE_RX</code>    | RX mode        |
| <code>USART_MODE_TX</code>    | TX mode        |
| <code>USART_MODE_TX_RX</code> | RX and TX mode |

***USART Parity***

|                                |             |
|--------------------------------|-------------|
| <code>USART_PARITY_NONE</code> | No parity   |
| <code>USART_PARITY_EVEN</code> | Even parity |

USART\_PARITY\_ODD Odd parity

**USART Request Parameters**

USART\_RXDATA\_FLUSH\_REQUEST Receive Data flush Request

USART\_TXDATA\_FLUSH\_REQUEST Transmit data flush Request

**USART Number of Stop Bits**

USART\_STOPBITS\_0\_5 USART frame with 0.5 stop bit

USART\_STOPBITS\_1 USART frame with 1 stop bit

USART\_STOPBITS\_1\_5 USART frame with 1.5 stop bits

USART\_STOPBITS\_2 USART frame with 2 stop bits

## 57 HAL USART Extension Driver

### 57.1 USARTEx Firmware driver defines

#### 57.1.1 USARTEx

##### *USARTEx Word Length*

USART\_WORDLENGTH\_8B 8-bit long USART frame

USART\_WORDLENGTH\_9B 9-bit long USART frame

## 58 HAL WWDG Generic Driver

### 58.1 WWDG Firmware driver registers structures

#### 58.1.1 WWDG\_InitTypeDef

##### Data Fields

- *uint32\_t Prescaler*
- *uint32\_t Window*
- *uint32\_t Counter*
- *uint32\_t EWIMode*

##### Field Documentation

- ***uint32\_t WWDG\_InitTypeDef::Prescaler***  
Specifies the prescaler value of the WWDG. This parameter can be a value of [WWDG\\_Prescaler](#)
- ***uint32\_t WWDG\_InitTypeDef::Window***  
Specifies the WWDG window value to be compared to the downcounter. This parameter must be a number Min\_Data = 0x40 and Max\_Data = 0x7FU
- ***uint32\_t WWDG\_InitTypeDef::Counter***  
Specifies the WWDG free-running downcounter value. This parameter must be a number between Min\_Data = 0x40 and Max\_Data = 0x7FU
- ***uint32\_t WWDG\_InitTypeDef::EWIMode***  
Specifies if WWDG Early Wakeup Interupt is enable or not. This parameter can be a value of [WWDG\\_EWI\\_Mode](#)

#### 58.1.2 WWDG\_HandleTypeDef

##### Data Fields

- *WWDG\_TypeDef \* Instance*
- *WWDG\_InitTypeDef Init*

##### Field Documentation

- ***WWDG\_TypeDef\* WWDG\_HandleTypeDef::Instance***  
Register base address
- ***WWDG\_InitTypeDef WWDG\_HandleTypeDef::Init***  
WWDG required parameters

### 58.2 WWDG Firmware driver API description

#### 58.2.1 WWDG specific features

Once enabled the WWDG generates a system reset on expiry of a programmed time period, unless the program refreshes the counter (T[6U;0] downcounter) before reaching 0x3F value (i.e. a reset is generated when the counter value rolls over from 0x40 to 0x3FU).

- An MCU reset is also generated if the counter value is refreshed before the counter has reached the refresh window value. This implies that the counter must be refreshed in a limited window.
- Once enabled the WWDG cannot be disabled except by a system reset.

- WWDRST flag in RCC\_CSR register informs when a WWDG reset has occurred (check available with \_\_HAL\_RCC\_GET\_FLAG(RCC\_FLAG\_WWDGRST)).
- The WWDG downcounter input clock is derived from the APB clock divided by a programmable prescaler.
- WWDG downcounter clock (Hz) = PCLK1 / (4096U \* Prescaler)
- WWDG timeout (ms) = (1000U \* (T[5U;0] + 1U)) / (WWDG downcounter clock) where T[5U;0] are the lowest 6 bits of downcounter.
- WWDG Counter refresh is allowed between the following limits :
  - min time (ms) = (1000U \* (T[5U;0] - Window)) / (WWDG downcounter clock)
  - max time (ms) = (1000U \* (T[5U;0] - 0x40U)) / (WWDG downcounter clock)
- Min-max timeout value @42 MHz(PCLK1): ~97.5 us / ~49.9 ms
- The Early Wakeup Interrupt (EWI) can be used if specific safety operations or data logging must be performed before the actual reset is generated. When the downcounter reaches the value 0x40U, an EWI interrupt is generated and the corresponding interrupt service routine (ISR) can be used to trigger specific actions (such as communications or data logging), before resetting the device. In some applications, the EWI interrupt can be used to manage a software system check and/or system recovery/graceful degradation, without generating a WWDG reset. In this case, the corresponding interrupt service routine (ISR) should reload the WWDG counter to avoid the WWDG reset, then trigger the required actions. Note:When the EWI interrupt cannot be served, e.g. due to a system lock in a higher priority task, the WWDG reset will eventually be generated.
- Debug mode : When the microcontroller enters debug mode (core halted), the WWDG counter either continues to work normally or stops, depending on DBG\_WWDG\_STOP configuration bit in DBG module, accessible through \_\_HAL\_DBGMCU\_FREEZE\_WWDG() and \_\_HAL\_DBGMCU\_UNFREEZE\_WWDG() macros

### 58.2.2 How to use this driver

- Enable WWDG APB1 clock using \_\_HAL\_RCC\_WWDG\_CLK\_ENABLE().
- Set the WWDG prescaler, refresh window, counter value and Early Wakeup Interrupt mode using using HAL\_WWDG\_Init() function. This enables WWDG peripheral and the downcounter starts downcounting from given counter value. Init function can be called again to modify all watchdog parameters, however if EWI mode has been set once, it can't be clear until next reset.
- The application program must refresh the WWDG counter at regular intervals during normal operation to prevent an MCU reset using HAL\_WWDG\_Refresh() function. This operation must occur only when the counter is lower than the window value already programmed.
- if Early Wakeup Interrupt mode is enable an interrupt is generated when the counter reaches 0x40. User can add his own code in weak function HAL\_WWDG\_EarlyWakeupCallback().

### WWDG HAL driver macros list

Below the list of most used macros in WWDG HAL driver.

- \_\_HAL\_WWDG\_GET\_IT\_SOURCE: Check the selected WWDG's interrupt source.
- \_\_HAL\_WWDG\_GET\_FLAG: Get the selected WWDG's flag status.
- \_\_HAL\_WWDG\_CLEAR\_FLAG: Clear the WWDG's pending flags.

### 58.2.3 Initialization and Configuration functions

This section provides functions allowing to:

- Initialize and start the WWDG according to the specified parameters in the WWDG\_InitTypeDef of associated handle.
- Initialize the WWDG MSP.

This section contains the following APIs:

- [\*HAL\\_WWDG\\_Init\(\)\*](#)
- [\*HAL\\_WWDG\\_MspInit\(\)\*](#)

### 58.2.4 IO operation functions

This section provides functions allowing to:

- Refresh the WWDG.
- Handle WWDG interrupt request and associated function callback.

This section contains the following APIs:

- [\*HAL\\_WWDG\\_Refresh\(\)\*](#)
- [\*HAL\\_WWDG\\_IRQHandler\(\)\*](#)
- [\*HAL\\_WWDG\\_EarlyWakeupCallback\(\)\*](#)

### 58.2.5 Detailed description of functions

#### **HAL\_WWDG\_Init**

|                      |                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_WWDG_Init<br/>(WWDG_HandleTypeDef * hwdg)</b>                                                                                                                |
| Function description | Initialize the WWDG according to the specified.                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hwdg:</b> pointer to a WWDG_HandleTypeDef structure that contains the configuration information for the specified WWDG module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                |

#### **HAL\_WWDG\_MspInit**

|                      |                                                                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_WWDG_MspInit (WWDG_HandleTypeDef * hwdg)</b>                                                                                                                                                             |
| Function description | Initialize the WWDG MSP.                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hwdg:</b> pointer to a WWDG_HandleTypeDef structure that contains the configuration information for the specified WWDG module.</li> </ul>                                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• When rewriting this function in user file, mechanism may be added to avoid multiple initialize when HAL_WWDG_Init function is called again to change parameters.</li> </ul> |

**HAL\_WWDG\_Refresh**

|                      |                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>HAL_StatusTypeDef HAL_WWDG_Refresh<br/>(WWDG_HandleTypeDef * hwdg)</b>                                                                                                             |
| Function description | Refresh the WWDG.                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hwdg:</b> pointer to a WWDG_HandleTypeDef structure that contains the configuration information for the specified WWDG module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>HAL:</b> status</li> </ul>                                                                                                                |

**HAL\_WWDG\_IRQHandler**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_WWDG_IRQHandler (WWDG_HandleTypeDef * hwdg)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Handle WWDG interrupt request.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hwdg:</b> pointer to a WWDG_HandleTypeDef structure that contains the configuration information for the specified WWDG module.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• The Early Wakeup Interrupt (EWI) can be used if specific safety operations or data logging must be performed before the actual reset is generated. The EWI interrupt is enabled by calling HAL_WWDG_Init function with EWIMode set to WWDG_EWI_ENABLE. When the downcounter reaches the value 0x40, and EWI interrupt is generated and the corresponding Interrupt Service Routine (ISR) can be used to trigger specific actions (such as communications or data logging), before resetting the device.</li> </ul> |

**HAL\_WWDG\_EarlyWakeupCallback**

|                      |                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void HAL_WWDG_EarlyWakeupCallback<br/>(WWDG_HandleTypeDef * hwdg)</b>                                                                                                              |
| Function description | WWDG Early Wakeup callback.                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>hwdg:</b> pointer to a WWDG_HandleTypeDef structure that contains the configuration information for the specified WWDG module.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                       |

## 58.3 WWDG Firmware driver defines

### 58.3.1 WWDG

***WWDG Early Wakeup Interrupt Mode***

**WWDG\_EWI\_DISABLE**    EWI Disable  
**WWDG\_EWI\_ENABLE**    EWI Enable

**WWDG Exported Macros**`_HAL_WWDG_ENABLE`**Description:**

- Enable the WWDG peripheral.

**Parameters:**

- `_HANDLE_`: WWDG handle

**Return value:**

- None

`_HAL_WWDG_ENABLE_IT`**Description:**

- Enable the WWDG early wakeup interrupt.

**Parameters:**

- `_HANDLE_`: WWDG handle
- `_INTERRUPT_`: specifies the interrupt to enable. This parameter can be one of the following values:
  - `WWDG_IT_EWI`: Early wakeup interrupt

**Return value:**

- None

**Notes:**

- Once enabled this interrupt cannot be disabled except by a system reset.

`_HAL_WWDG_GET_IT`**Description:**

- Check whether the selected WWDG interrupt has occurred or not.

**Parameters:**

- `_HANDLE_`: WWDG handle
- `_INTERRUPT_`: specifies the it to check. This parameter can be one of the following values:
  - `WWDG_FLAG_EWIF`: Early wakeup interrupt IT

**Return value:**

- The: new state of `WWDG_FLAG` (SET or RESET).

`_HAL_WWDG_CLEAR_IT`**Description:**

- Clear the WWDG interrupt pending bits.

**Parameters:**

- `_HANDLE_`: WWDG handle
- `_INTERRUPT_`: specifies the interrupt pending bit to clear. This parameter can be one of the following values:
  - `WWDG_FLAG_EWIF`: Early wakeup interrupt flag

[\\_\\_HAL\\_WWDG\\_GET\\_FLAG](#)**Description:**

- Check whether the specified WWdg flag is set or not.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): WWdg handle
- [\\_\\_FLAG\\_\\_](#): specifies the flag to check. This parameter can be one of the following values:
  - [WWDG\\_FLAG\\_EWIF](#): Early wakeup interrupt flag

**Return value:**

- The: new state of [WWDG\\_FLAG](#) (SET or RESET).

[\\_\\_HAL\\_WWDG\\_CLEAR\\_FLAG](#)**Description:**

- Clear the WWdg's pending flags.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): WWdg handle
- [\\_\\_FLAG\\_\\_](#): specifies the flag to clear. This parameter can be one of the following values:
  - [WWDG\\_FLAG\\_EWIF](#): Early wakeup interrupt flag

**Return value:**

- None

[\\_\\_HAL\\_WWDG\\_GET\\_IT\\_SOURCE](#)**Description:**

- Check whether the specified WWdg interrupt source is enabled or not.

**Parameters:**

- [\\_\\_HANDLE\\_\\_](#): WWdg Handle.
- [\\_\\_INTERRUPT\\_\\_](#): specifies the WWdg interrupt source to check. This parameter can be one of the following values:
  - [WWDG\\_IT\\_EWI](#): Early Wakeup Interrupt

**Return value:**

- state: of [\\_\\_INTERRUPT\\_\\_](#) (TRUE or FALSE).

***WWdg Flag definition***

[WWDG\\_FLAG\\_EWIF](#) Early wakeup interrupt flag

***WWdg Interrupt definition***

[WWDG\\_IT\\_EWI](#) Early wakeup interrupt

***WWDG Prescaler***

- WWDG\_PRESCALER\_1    WWDG counter clock = (PCLK1/4096U)/1U
- WWDG\_PRESCALER\_2    WWDG counter clock = (PCLK1/4096U)/2U
- WWDG\_PRESCALER\_4    WWDG counter clock = (PCLK1/4096U)/4U
- WWDG\_PRESCALER\_8    WWDG counter clock = (PCLK1/4096U)/8U

## 59 LL ADC Generic Driver

### 59.1 ADC Firmware driver registers structures

#### 59.1.1 LL\_ADC\_CommonInitTypeDef

##### Data Fields

- *uint32\_t CommonClock*
- *uint32\_t Multimode*
- *uint32\_t MultiDMATransfer*
- *uint32\_t MultiTwoSamplingDelay*

##### Field Documentation

- *uint32\_t LL\_ADC\_CommonInitTypeDef::CommonClock*

Set parameter common to several ADC: Clock source and prescaler. This parameter can be a value of [ADC\\_LL\\_EC\\_COMMON\\_CLOCK\\_SOURCE](#)

**Note:**On this STM32 serie, if ADC group injected is used, some clock ratio constraints between ADC clock and AHB clock must be respected. Refer to reference manual.

This feature can be modified afterwards using unitary function

[LL\\_ADC\\_SetCommonClock\(\)](#).

- *uint32\_t LL\_ADC\_CommonInitTypeDef::Multimode*

Set ADC multimode configuration to operate in independent mode or multimode (for devices with several ADC instances). This parameter can be a value of [ADC\\_LL\\_EC\\_MULTI\\_MODE](#)

This feature can be modified afterwards using unitary function [LL\\_ADC\\_SetMultimode\(\)](#).

- *uint32\_t LL\_ADC\_CommonInitTypeDef::MultiDMATransfer*

Set ADC multimode conversion data transfer: no transfer or transfer by DMA. This parameter can be a value of [ADC\\_LL\\_EC\\_MULTI\\_DMA\\_TRANSFER](#)

This feature can be modified afterwards using unitary function [LL\\_ADC\\_SetMultiDMATransfer\(\)](#).

- *uint32\_t LL\_ADC\_CommonInitTypeDef::MultiTwoSamplingDelay*

Set ADC multimode delay between 2 sampling phases. This parameter can be a value of [ADC\\_LL\\_EC\\_MULTI\\_TWOSMP\\_DELAY](#)

This feature can be modified afterwards using unitary function [LL\\_ADC\\_SetMultiTwoSamplingDelay\(\)](#).

#### 59.1.2 LL\_ADC\_InitTypeDef

##### Data Fields

- *uint32\_t Resolution*
- *uint32\_t DataAlignment*
- *uint32\_t LowPowerMode*

##### Field Documentation

- *uint32\_t LL\_ADC\_InitTypeDef::Resolution*

Set ADC resolution. This parameter can be a value of

[ADC\\_LL\\_EC\\_RESOLUTION](#)

This feature can be modified afterwards using unitary function [LL\\_ADC\\_SetResolution\(\)](#).

- *uint32\_t LL\_ADC\_InitTypeDef::DataAlignment*

Set ADC conversion data alignment. This parameter can be a value of

[ADC\\_LL\\_EC\\_DATA\\_ALIGN](#)

This feature can be modified afterwards using unitary function [LL\\_ADC\\_SetDataAlignment\(\)](#).

- ***uint32\_t LL\_ADC\_InitTypeDef::LowPowerMode***  
Set ADC low power mode. This parameter can be a value of ***ADC\_LL\_EC\_LP\_MODE***This feature can be modified afterwards using unitary function ***LL\_ADC\_SetLowPowerMode()***.

### 59.1.3 LL\_ADC\_REG\_InitTypeDef

#### Data Fields

- ***uint32\_t TriggerSource***
- ***uint32\_t SequencerLength***
- ***uint32\_t SequencerDiscont***
- ***uint32\_t ContinuousMode***
- ***uint32\_t DMATransfer***
- ***uint32\_t Overrun***

#### Field Documentation

- ***uint32\_t LL\_ADC\_REG\_InitTypeDef::TriggerSource***  
Set ADC group regular conversion trigger source: internal (SW start) or from external IP (timer event, external interrupt line). This parameter can be a value of ***ADC\_LL\_EC\_REG\_TRIGGER\_SOURCE***  
**Note:**On this STM32 serie, setting trigger source to external trigger also set trigger polarity to rising edge (default setting for compatibility with some ADC on other STM32 families having this setting set by HW default value). In case of need to modify trigger edge, use function ***LL\_ADC\_REG\_SetTriggerEdge()***. This feature can be modified afterwards using unitary function ***LL\_ADC\_REG\_SetTriggerSource()***.
- ***uint32\_t LL\_ADC\_REG\_InitTypeDef::SequencerLength***  
Set ADC group regular sequencer length. This parameter can be a value of ***ADC\_LL\_EC\_REG\_SEQ\_SCAN\_LENGTH***This feature can be modified afterwards using unitary function ***LL\_ADC\_REG\_SetSequencerLength()***.
- ***uint32\_t LL\_ADC\_REG\_InitTypeDef::SequencerDiscont***  
Set ADC group regular sequencer discontinuous mode: sequence subdivided and scan conversions interrupted every selected number of ranks. This parameter can be a value of ***ADC\_LL\_EC\_REG\_SEQ\_DISCONT\_MODE***  
**Note:**This parameter has an effect only if group regular sequencer is enabled (scan length of 2 ranks or more). This feature can be modified afterwards using unitary function ***LL\_ADC\_REG\_SetSequencerDiscont()***.
- ***uint32\_t LL\_ADC\_REG\_InitTypeDef::ContinuousMode***  
Set ADC continuous conversion mode on ADC group regular, whether ADC conversions are performed in single mode (one conversion per trigger) or in continuous mode (after the first trigger, following conversions launched successively automatically). This parameter can be a value of ***ADC\_LL\_EC\_REG\_CONTINUOUS\_MODE*** Note: It is not possible to enable both ADC group regular continuous mode and discontinuous mode.This feature can be modified afterwards using unitary function ***LL\_ADC\_REG\_SetContinuousMode()***.
- ***uint32\_t LL\_ADC\_REG\_InitTypeDef::DMATransfer***  
Set ADC group regular conversion data transfer: no transfer or transfer by DMA, and DMA requests mode. This parameter can be a value of ***ADC\_LL\_EC\_REG\_DMA\_TRANSFER***This feature can be modified afterwards using unitary function ***LL\_ADC\_REG\_SetDMATransfer()***.
- ***uint32\_t LL\_ADC\_REG\_InitTypeDef::Overrun***  
Set ADC group regular behavior in case of overrun: data preserved or overwritten. This parameter can be a value of ***ADC\_LL\_EC\_REG\_OVR\_DATA\_BEHAVIOR***This feature can be modified afterwards using unitary function ***LL\_ADC\_REG\_SetOverrun()***.

### 59.1.4 LL\_ADC\_INJ\_InitTypeDef

#### Data Fields

- *uint32\_t TriggerSource*
- *uint32\_t SequencerLength*
- *uint32\_t SequencerDiscont*
- *uint32\_t TrigAuto*

#### Field Documentation

- *uint32\_t LL\_ADC\_INJ\_InitTypeDef::TriggerSource*

Set ADC group injected conversion trigger source: internal (SW start) or from external IP (timer event, external interrupt line). This parameter can be a value of [\*\*ADC\\_LL\\_EC\\_INJ\\_TRIGGER\\_SOURCE\*\*](#)

**Note:**On this STM32 serie, setting trigger source to external trigger also set trigger polarity to rising edge (default setting for compatibility with some ADC on other STM32 families having this setting set by HW default value). In case of need to modify trigger edge, use function [\*\*LL\\_ADC\\_INJ\\_SetTriggerEdge\(\)\*\*](#). This feature can be modified afterwards using unitary function [\*\*LL\\_ADC\\_INJ\\_SetTriggerSource\(\)\*\*](#).

- *uint32\_t LL\_ADC\_INJ\_InitTypeDef::SequencerLength*

Set ADC group injected sequencer length. This parameter can be a value of [\*\*ADC\\_LL\\_EC\\_INJ\\_SEQ\\_SCAN\\_LENGTH\*\*](#)This feature can be modified afterwards using unitary function [\*\*LL\\_ADC\\_INJ\\_SetSequencerLength\(\)\*\*](#).

- *uint32\_t LL\_ADC\_INJ\_InitTypeDef::SequencerDiscont*

Set ADC group injected sequencer discontinuous mode: sequence subdivided and scan conversions interrupted every selected number of ranks. This parameter can be a value of [\*\*ADC\\_LL\\_EC\\_INJ\\_SEQ\\_DISCONT\\_MODE\*\*](#)

**Note:**This parameter has an effect only if group injected sequencer is enabled (scan length of 2 ranks or more). This feature can be modified afterwards using unitary function [\*\*LL\\_ADC\\_INJ\\_SetSequencerDiscont\(\)\*\*](#).

- *uint32\_t LL\_ADC\_INJ\_InitTypeDef::TrigAuto*

Set ADC group injected conversion trigger: independent or from ADC group regular. This parameter can be a value of [\*\*ADC\\_LL\\_EC\\_INJ\\_TRIG\\_AUTO\*\*](#) Note: This parameter must be set to set to independent trigger if injected trigger source is set to an external trigger.This feature can be modified afterwards using unitary function [\*\*LL\\_ADC\\_INJ\\_SetTrigAuto\(\)\*\*](#).

## 59.2 ADC Firmware driver API description

### 59.2.1 Detailed description of functions

#### LL\_ADC\_DMA\_GetRegAddr

Function name [\\_\\_STATIC\\_INLINE uint32\\_t LL\\_ADC\\_DMA\\_GetRegAddr\(ADC\\_TypeDef \\* ADCx, uint32\\_t Register\)](#)

Function description Function to help to configure DMA transfer from ADC: retrieve the ADC register address from ADC instance and a list of ADC registers intended to be used (most commonly) with DMA transfer.

Parameters
 

- **ADCx:** ADC instance
- **Register:** This parameter can be one of the following values:  
(1) Available on devices with several ADC instances.
  - [\*\*LL\\_ADC\\_DMA\\_REG\\_REGULAR\\_DATA\*\*](#)
  - [\*\*LL\\_ADC\\_DMA\\_REG\\_REGULAR\\_DATA\\_MULTI\*\*](#) (1)

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>ADC:</b> register address</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>These ADC registers are data registers: when ADC conversion data is available in ADC data registers, ADC generates a DMA transfer request.</li> <li>This macro is intended to be used with LL DMA driver, refer to function "LL_DMA_ConfigAddresses()". Example:<br/>LL_DMA_ConfigAddresses(DMA1, LL_DMA_CHANNEL_1,<br/>LL_ADC_DMA_GetRegAddr(ADC1),<br/>LL_ADC_DMA_REG_REGULAR_DATA), (uint32_t)&amp;&lt; array<br/>or variable &gt;,<br/>LL_DMA_DIRECTION_PERIPH_TO_MEMORY);</li> <li>For devices with several ADC: in multimode, some devices use a different data register outside of ADC instance scope (common data register). This macro manages this register difference, only ADC instance has to be set as parameter.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>DR RDATA LL_ADC_DMA_GetRegAddr</li> <li>CDR RDATA_MST LL_ADC_DMA_GetRegAddr</li> <li>CDR RDATA_SLV LL_ADC_DMA_GetRegAddr</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

## LL\_ADC\_SetCommonClock

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_ADC_SetCommonClock<br/>(ADC_Common_TypeDef * ADCxy_COMMON, uint32_t<br/>CommonClock)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Set parameter common to several ADC: Clock source and prescaler.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>_LL_ADC_COMMON_INSTANCE()</code>)</li> <li><b>CommonClock:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_ADC_CLOCK_SYNC_PCLK_DIV1</code></li> <li>- <code>LL_ADC_CLOCK_SYNC_PCLK_DIV2</code></li> <li>- <code>LL_ADC_CLOCK_SYNC_PCLK_DIV4</code></li> <li>- <code>LL_ADC_CLOCK_ASYNC_DIV1</code></li> </ul> </li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>On this STM32 serie, if ADC group injected is used, some clock ratio constraints between ADC clock and AHB clock must be respected. Refer to reference manual.</li> <li>On this STM32 serie, setting of this feature is conditioned to ADC state: All ADC instances of the ADC common group must be disabled. This check can be done with function <code>LL_ADC_IsEnabled()</code> for each ADC instance or by using helper macro helper macro <code>_LL_ADC_IS_ENABLED_ALL_COMMON_INSTANCE()</code>.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CCR CKMODE LL_ADC_SetCommonClock</li> <li>CCR PRESC LL_ADC_SetCommonClock</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                             |

**LL\_ADC\_GetCommonClock**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_GetCommonClock(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                                                                                                                                                                                                   |
| Function description                              | Get parameter common to several ADC: Clock source and prescaler.                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul>                                                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:             <ul style="list-style-type: none"> <li>- <code>LL_ADC_CLOCK_SYNC_PCLK_DIV1</code></li> <li>- <code>LL_ADC_CLOCK_SYNC_PCLK_DIV2</code></li> <li>- <code>LL_ADC_CLOCK_SYNC_PCLK_DIV4</code></li> <li>- <code>LL_ADC_CLOCK_ASYNC_DIV1</code></li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR CKMODE <code>LL_ADC_SetCommonClock</code></li> <li>• CCR PRESC <code>LL_ADC_SetCommonClock</code></li> </ul>                                                                                                                                                                                                                        |

**LL\_ADC\_SetCommonPathInternalCh**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_SetCommonPathInternalCh(ADC_Common_TypeDef * ADCxy_COMMON, uint32_t PathInternal)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description | Set parameter common to several ADC: measurement path to internal channels (VrefInt, temperature sensor, ...).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> <li>• <b>PathInternal:</b> This parameter can be a combination of the following values:             <ul style="list-style-type: none"> <li>- <code>LL_ADC_PATH_INTERNAL_NONE</code></li> <li>- <code>LL_ADC_PATH_INTERNAL_VREFINT</code></li> <li>- <code>LL_ADC_PATH_INTERNAL_TEMPSENSOR</code></li> <li>- <code>LL_ADC_PATH_INTERNAL_VBAT</code></li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• One or several values can be selected. Example:<br/><code>(LL_ADC_PATH_INTERNAL_VREFINT   LL_ADC_PATH_INTERNAL_TEMPSENSOR)</code></li> <li>• Stabilization time of measurement path to internal channel:<br/>After enabling internal paths, before starting ADC conversion, a delay is required for internal voltage reference and temperature sensor stabilization time. Refer to device datasheet. Refer to literal <code>LL_ADC_DELAY_VREFINT_STAB_US</code>. Refer to literal <code>LL_ADC_DELAY_TEMPSENSOR_STAB_US</code>.</li> <li>• ADC internal channel sampling time constraint: For ADC conversion of internal channels, a sampling time minimum value is required. Refer to device datasheet.</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: All ADC instances of the ADC common group must be disabled. This check can be done with function</li> </ul> |

|                                             |                                                                                                                                                                                                 |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CCR VREFEN LL_ADC_SetCommonPathInternalCh</li> <li>CCR TSEN LL_ADC_SetCommonPathInternalCh</li> <li>CCR VBATEN LL_ADC_SetCommonPathInternalCh</li> </ul> |
|                                             | LL_ADC_IsEnabled() for each ADC instance or by using helper macro helper macro<br>__LL_ADC_IS_ENABLED_ALL_COMMON_INSTANCE().                                                                    |

### LL\_ADC\_GetCommonPathInternalCh

|                                             |                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_GetCommonPathInternalCh<br/>(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                                                                                                                              |
| Function description                        | Get parameter common to several ADC: measurement path to internal channels (VrefInt, temperature sensor, ...).                                                                                                                                                                                                                |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro __LL_ADC_COMMON_INSTANCE() )</li> </ul>                                                                                                                                   |
| Return values                               | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_PATH_INTERNAL_NONE</li> <li>- LL_ADC_PATH_INTERNAL_VREFINT</li> <li>- LL_ADC_PATH_INTERNAL_TEMPSENSOR</li> <li>- LL_ADC_PATH_INTERNAL_VBAT</li> </ul> </li> </ul> |
| Notes                                       | <ul style="list-style-type: none"> <li>One or several values can be selected. Example:<br/>(LL_ADC_PATH_INTERNAL_VREFINT   LL_ADC_PATH_INTERNAL_TEMPSENSOR)</li> </ul>                                                                                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CCR VREFEN LL_ADC_GetCommonPathInternalCh</li> <li>CCR TSEN LL_ADC_GetCommonPathInternalCh</li> <li>CCR VBATEN LL_ADC_GetCommonPathInternalCh</li> </ul>                                                                                                                               |

### LL\_ADC\_SetCalibrationFactor

|                      |                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_SetCalibrationFactor<br/>(ADC_TypeDef * ADCx, uint32_t SingleDiff, uint32_t CalibrationFactor)</code>                                                                                                                                                                                                                                                            |
| Function description | Set ADC calibration factor in the mode single-ended or differential (for devices with differential mode available).                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> <li><b>SingleDiff:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_SINGLE_ENDED</li> <li>- LL_ADC_DIFFERENTIAL_ENDED</li> <li>- LL_ADC_BOTH_SINGLE_DIFF_ENDED</li> </ul> </li> <li><b>CalibrationFactor:</b> Value between Min_Data=0x00 and Max_Data=0x7F</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>This function is intended to set calibration parameters without having to perform a new calibration using LL_ADC_StartCalibration().</li> </ul>                                                                                                                                                                                                             |

- For devices with differential mode available: Calibration of offset is specific to each of single-ended and differential modes (calibration factor must be specified for each of these differential modes, if used afterwards and if the application requires their calibration).
- In case of setting calibration factors of both modes single ended and differential (parameter LL\_ADC\_BOTH\_SINGLE\_DIFF\_ENDED): both calibration factors must be concatenated. To perform this processing, use helper macro \_\_LL\_ADC\_CALIB\_FACTOR\_SINGLE\_DIFF().
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be enabled, without calibration on going, without conversion on going on group regular.

Reference Manual to  
LL API cross  
reference:

- CALFACT CALFACT\_S LL\_ADC\_SetCalibrationFactor
- CALFACT CALFACT\_D LL\_ADC\_SetCalibrationFactor

### **LL\_ADC\_GetCalibrationFactor**

|                                                   |                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_ADC_GetCalibrationFactor(ADC_TypeDef * ADCx, uint32_t SingleDiff)</code></b>                                                                                                                                                                                                  |
| Function description                              | Get ADC calibration factor in the mode single-ended or differential (for devices with differential mode available).                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>SingleDiff:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_ADC_SINGLE_ENDED</li> <li>– LL_ADC_DIFFERENTIAL_ENDED</li> </ul> </li> </ul>                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x7F</li> </ul>                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• Calibration factors are set by hardware after performing a calibration run using function LL_ADC_StartCalibration().</li> <li>• For devices with differential mode available: Calibration of offset is specific to each of single-ended and differential modes</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CALFACT CALFACT_S LL_ADC_SetCalibrationFactor</li> <li>• CALFACT CALFACT_D LL_ADC_SetCalibrationFactor</li> </ul>                                                                                                                                                         |

### **LL\_ADC\_SetResolution**

|                      |                                                                                                                                                                                                                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_ADC_SetResolution(ADC_TypeDef * ADCx, uint32_t Resolution)</code></b>                                                                                                                                                                                                     |
| Function description | Set ADC resolution.                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Resolution:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_ADC_RESOLUTION_12B</li> <li>– LL_ADC_RESOLUTION_10B</li> <li>– LL_ADC_RESOLUTION_8B</li> </ul> </li> </ul> |

---

|                                                   |                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | – LL_ADC_RESOLUTION_6B                                                                                                                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR RES LL_ADC_SetResolution</li> </ul>                                                                                                                                               |

### LL\_ADC\_GetResolution

|                                                   |                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_ADC_GetResolution(ADC_TypeDef * ADCx)</code></b>                                                                                                                                                                                                                 |
| Function description                              | Get ADC resolution.                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_ADC_RESOLUTION_12B</li> <li>– LL_ADC_RESOLUTION_10B</li> <li>– LL_ADC_RESOLUTION_8B</li> <li>– LL_ADC_RESOLUTION_6B</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR RES LL_ADC_GetResolution</li> </ul>                                                                                                                                                                                                                    |

### LL\_ADC\_SetDataAlignment

|                                                   |                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_ADC_SetDataAlignment(ADC_TypeDef * ADCx, uint32_t DataAlignment)</code></b>                                                                                                                                                                                                                  |
| Function description                              | Set ADC conversion data alignment.                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>DataAlignment:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_ADC_DATA_ALIGN_RIGHT</li> <li>– LL_ADC_DATA_ALIGN_LEFT</li> </ul> </li> </ul>                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• Refer to reference manual for alignments formats dependencies to ADC resolutions.</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR ALIGN LL_ADC_SetDataAlignment</li> </ul>                                                                                                                                                                                                                                       |

**LL\_ADC\_GetDataAlignment**

|                                                   |                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_ADC_GetDataAlignment(ADC_TypeDef * ADCx)</code></b>                                                                                                                                                   |
| Function description                              | Get ADC conversion data alignment.                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_ADC_DATA_ALIGN_RIGHT</li> <li>– LL_ADC_DATA_ALIGN_LEFT</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• Refer to reference manual for alignments formats dependencies to ADC resolutions.</li> <li>• CFGR ALIGN LL_ADC_GetDataAlignment</li> </ul>                                                       |
| Reference Manual to<br>LL API cross<br>reference: |                                                                                                                                                                                                                                           |

**LL\_ADC\_SetLowPowerMode**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_ADC_SetLowPowerMode(ADC_TypeDef * ADCx, uint32_t LowPowerMode)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description | Set ADC low power mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>LowPowerMode:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_ADC_LP_MODE_NONE</li> <li>– LL_ADC_LP_AUTOWAIT</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Notes                | <ul style="list-style-type: none"> <li>• Description of ADC low power modes: ADC low power mode "auto wait": Dynamic low power mode, ADC conversions occurrences are limited to the minimum necessary in order to reduce power consumption. New ADC conversion starts only when the previous unitary conversion data (for ADC group regular) or previous sequence conversions data (for ADC group injected) has been retrieved by user software. In the meantime, ADC remains idle: does not performs any other conversion. This mode allows to automatically adapt the ADC conversions triggers to the speed of the software that reads the data. Moreover, this avoids risk of overrun for low frequency applications. How to use this low power mode: Do not use with interruption or DMA since these modes have to clear immediately the EOC flag to free the IRQ vector sequencer. Do use with polling: 1. Start conversion, 2. Later on, when conversion data is needed: poll for end of conversion to ensure that conversion is completed and retrieve ADC conversion data. This will trig another ADC conversion start. ADC low power mode "auto power-off" (feature available on this device if parameter LL_ADC_LP_MODE_AUTOOFF is available): the ADC automatically powers-off after a conversion and automatically wakes up when a new conversion is triggered (with startup time between trigger and start of sampling). This feature can</li> </ul> |

be combined with low power mode "auto wait".

- With ADC low power mode "auto wait", the ADC conversion data read is corresponding to previous ADC conversion start, independently of delay during which ADC was idle. Therefore, the ADC conversion data may be outdated: does not correspond to the current voltage level on the selected ADC channel.
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.
- CFGREG AUTDLY LL\_ADC\_SetLowPowerMode

Reference Manual to  
LL API cross  
reference:

### **LL\_ADC\_GetLowPowerMode**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_GetLowPowerMode(<br/>(ADC_TypeDef * ADCx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description | Get ADC low power mode:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Return values        | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_ADC_LP_MODE_NONE</li> <li>- LL_ADC_LP_AUTOWAIT</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>Description of ADC low power modes: ADC low power mode "auto wait": Dynamic low power mode, ADC conversions occurrences are limited to the minimum necessary in order to reduce power consumption. New ADC conversion starts only when the previous unitary conversion data (for ADC group regular) or previous sequence conversions data (for ADC group injected) has been retrieved by user software. In the meantime, ADC remains idle: does not perform any other conversion. This mode allows to automatically adapt the ADC conversions triggers to the speed of the software that reads the data. Moreover, this avoids risk of overrun for low frequency applications. How to use this low power mode: Do not use with interruption or DMA since these modes have to clear immediately the EOC flag to free the IRQ vector sequencer. Do use with polling: 1. Start conversion, 2. Later on, when conversion data is needed: poll for end of conversion to ensure that conversion is completed and retrieve ADC conversion data. This will trigger another ADC conversion start. ADC low power mode "auto power-off" (feature available on this device if parameter LL_ADC_LP_MODE_AUTOOFF is available): the ADC automatically powers-off after a conversion and automatically wakes up when a new conversion is triggered (with startup time between trigger and start of sampling). This feature can be combined with low power mode "auto wait".</li> <li>With ADC low power mode "auto wait", the ADC conversion data read is corresponding to previous ADC conversion start, independently of delay during which ADC was idle. Therefore, the ADC conversion data may be outdated: does not</li> </ul> |

correspond to the current voltage level on the selected ADC channel.

Reference Manual to  
LL API cross  
reference:

- CFGR AUTDLY LL\_ADC\_GetLowPowerMode

### LL\_ADC\_SetOffset

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_SetOffset (ADC_TypeDef *<br/>ADCx, uint32_t Offsety, uint32_t Channel, uint32_t<br/>OffsetLevel)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description | Set ADC selected offset number 1, 2, 3 or 4.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Offsety:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_ADC_OFFSET_1</li> <li>– LL_ADC_OFFSET_2</li> <li>– LL_ADC_OFFSET_3</li> <li>– LL_ADC_OFFSET_4</li> </ul> </li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>(1) On STM32F3, parameter available only on ADC instance: ADC1. <ul style="list-style-type: none"> <li>– LL_ADC_CHANNEL_0</li> <li>– LL_ADC_CHANNEL_1</li> <li>– LL_ADC_CHANNEL_2</li> <li>– LL_ADC_CHANNEL_3</li> <li>– LL_ADC_CHANNEL_4</li> <li>– LL_ADC_CHANNEL_5</li> <li>– LL_ADC_CHANNEL_6</li> <li>– LL_ADC_CHANNEL_7</li> <li>– LL_ADC_CHANNEL_8</li> <li>– LL_ADC_CHANNEL_9</li> <li>– LL_ADC_CHANNEL_10</li> <li>– LL_ADC_CHANNEL_11</li> <li>– LL_ADC_CHANNEL_12</li> <li>– LL_ADC_CHANNEL_13</li> <li>– LL_ADC_CHANNEL_14</li> <li>– LL_ADC_CHANNEL_15</li> <li>– LL_ADC_CHANNEL_16</li> <li>– LL_ADC_CHANNEL_17</li> <li>– LL_ADC_CHANNEL_18</li> <li>– LL_ADC_CHANNEL_VREFINT (5)</li> <li>– LL_ADC_CHANNEL_TEMPSENSOR (1)</li> <li>– LL_ADC_CHANNEL_VBAT (1)</li> <li>– LL_ADC_CHANNEL_VOPAMP1 (1)</li> <li>– LL_ADC_CHANNEL_VOPAMP2 (2)</li> <li>– LL_ADC_CHANNEL_VOPAMP3 (3)</li> <li>– LL_ADC_CHANNEL_VOPAMP4 (4)</li> </ul> </li> <li>(2) On STM32F3, parameter available only on ADC instance: ADC2.</li> <li>(3) On STM32F3, parameter available only on ADC instance: ADC3.</li> <li>(4) On STM32F3, parameter available only on ADC</li> </ul> </li> </ul> |

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                             | <ul style="list-style-type: none"> <li>instances: ADC4.</li> <li>(5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.</li> <li><b>OffsetLevel:</b> Value between Min_Data=0x000 and Max_Data=0xFFFF</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Notes                                       | <ul style="list-style-type: none"> <li>This function set the 2 items of offset configuration: ADC channel to which the offset programmed will be applied (independently of channel mapped on ADC group regular or group injected)Offset level (offset to be subtracted from the raw converted data).</li> <li>Caution: Offset format is dependent to ADC resolution: offset has to be left-aligned on bit 11, the LSB (right bits) are set to 0.</li> <li>This function enables the offset, by default. It can be forced to disable state using function LL_ADC_SetOffsetState().</li> <li>If a channel is mapped on several offsets numbers, only the offset with the lowest value is considered for the subtraction.</li> <li>On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>OFR1 OFFSET1_CH LL_ADC_SetOffset</li> <li>OFR1 OFFSET1_LL ADC_SetOffset</li> <li>OFR1 OFFSET1_EN LL_ADC_SetOffset</li> <li>OFR2 OFFSET2_CH LL_ADC_SetOffset</li> <li>OFR2 OFFSET2_LL ADC_SetOffset</li> <li>OFR2 OFFSET2_EN LL_ADC_SetOffset</li> <li>OFR3 OFFSET3_CH LL_ADC_SetOffset</li> <li>OFR3 OFFSET3_LL ADC_SetOffset</li> <li>OFR3 OFFSET3_EN LL_ADC_SetOffset</li> <li>OFR4 OFFSET4_CH LL_ADC_SetOffset</li> <li>OFR4 OFFSET4_LL ADC_SetOffset</li> <li>OFR4 OFFSET4_EN LL_ADC_SetOffset</li> </ul>                                                                                                                                                                                                                                                                                                                                                                   |

### LL\_ADC\_GetOffsetChannel

|                      |                                                                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>STATIC_INLINE uint32_t LL_ADC_GetOffsetChannel(ADC_TypeDef * ADCx, uint32_t Offsety)</b>                                                                                                                                                                                                                   |
| Function description | Get for the ADC selected offset number 1, 2, 3 or 4: Channel to which the offset programmed will be applied (independently of channel mapped on ADC group regular or group injected)                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> <li><b>Offsety:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_OFFSET_1</li> <li>- LL_ADC_OFFSET_2</li> <li>- LL_ADC_OFFSET_3</li> <li>- LL_ADC_OFFSET_4</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values: (1) On STM32F3, parameter available only on ADC instance: ADC1.</li> </ul>                                                                                                                                  |

- LL\_ADC\_CHANNEL\_0
- LL\_ADC\_CHANNEL\_1
- LL\_ADC\_CHANNEL\_2
- LL\_ADC\_CHANNEL\_3
- LL\_ADC\_CHANNEL\_4
- LL\_ADC\_CHANNEL\_5
- LL\_ADC\_CHANNEL\_6
- LL\_ADC\_CHANNEL\_7
- LL\_ADC\_CHANNEL\_8
- LL\_ADC\_CHANNEL\_9
- LL\_ADC\_CHANNEL\_10
- LL\_ADC\_CHANNEL\_11
- LL\_ADC\_CHANNEL\_12
- LL\_ADC\_CHANNEL\_13
- LL\_ADC\_CHANNEL\_14
- LL\_ADC\_CHANNEL\_15
- LL\_ADC\_CHANNEL\_16
- LL\_ADC\_CHANNEL\_17
- LL\_ADC\_CHANNEL\_18
- LL\_ADC\_CHANNEL\_VREFINT (5)
- LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
- LL\_ADC\_CHANNEL\_VBAT (1)
- LL\_ADC\_CHANNEL\_VOPAMP1 (1)
- LL\_ADC\_CHANNEL\_VOPAMP2 (2)
- LL\_ADC\_CHANNEL\_VOPAMP3 (3)
- LL\_ADC\_CHANNEL\_VOPAMP4 (4)
- (2) On STM32F3, parameter available only on ADC instance: ADC2.
- (3) On STM32F3, parameter available only on ADC instance: ADC3.
- (4) On STM32F3, parameter available only on ADC instances: ADC4.
- (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to Vrefint at the same time.
- (1, 2, 3, 4, 5) For ADC channel read back from ADC register, comparison with internal channel parameter to be done using helper macro  
`_LL_ADC_CHANNEL_INTERNAL_TO_EXTERNAL()`.

**Notes**

- Usage of the returned channel number: To reinject this channel into another function LL\_ADC\_xxx: the returned channel number is only partly formatted on definition of literals LL\_ADC\_CHANNEL\_x. Therefore, it has to be compared with parts of literals LL\_ADC\_CHANNEL\_x or using helper macro  
`_LL_ADC_CHANNEL_TO_DECIMAL_NB()`. Then the selected literal LL\_ADC\_CHANNEL\_x can be used as parameter for another function. To get the channel number in decimal format: process the returned value with the helper macro `_LL_ADC_CHANNEL_TO_DECIMAL_NB()`.

**Reference Manual to  
LL API cross  
reference:**

- OFR1 OFFSET1\_CH LL\_ADC\_GetOffsetChannel
- OFR2 OFFSET2\_CH LL\_ADC\_GetOffsetChannel
- OFR3 OFFSET3\_CH LL\_ADC\_GetOffsetChannel

- OFR4 OFFSET4\_CH LL\_ADC\_GetOffsetChannel

### LL\_ADC\_GetOffsetLevel

|                                                   |                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_GetOffsetLevel(ADC_TypeDef * ADCx, uint32_t Offsety)</code>                                                                                                                                                                                                                 |
| Function description                              | Get for the ADC selected offset number 1, 2, 3 or 4: Offset level (offset to be subtracted from the raw converted data).                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Offsety:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_OFFSET_1</li> <li>- LL_ADC_OFFSET_2</li> <li>- LL_ADC_OFFSET_3</li> <li>- LL_ADC_OFFSET_4</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x000 and Max_Data=0xFFFF</li> </ul>                                                                                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• Caution: Offset format is dependent to ADC resolution: offset has to be left-aligned on bit 11, the LSB (right bits) are set to 0.</li> </ul>                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OFR1 OFFSET1 LL_ADC_GetOffsetLevel</li> <li>• OFR2 OFFSET2 LL_ADC_GetOffsetLevel</li> <li>• OFR3 OFFSET3 LL_ADC_GetOffsetLevel</li> <li>• OFR4 OFFSET4 LL_ADC_GetOffsetLevel</li> </ul>                                                                                  |

### LL\_ADC\_SetOffsetState

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_SetOffsetState(ADC_TypeDef * ADCx, uint32_t Offsety, uint32_t OffsetState)</code>                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Set for the ADC selected offset number 1, 2, 3 or 4: force offset state disable or enable without modifying offset channel or offset value.                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Offsety:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_OFFSET_1</li> <li>- LL_ADC_OFFSET_2</li> <li>- LL_ADC_OFFSET_3</li> <li>- LL_ADC_OFFSET_4</li> </ul> </li> <li>• <b>OffsetState:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_OFFSET_DISABLE</li> <li>- LL_ADC_OFFSET_ENABLE</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Notes                | <ul style="list-style-type: none"> <li>• This function should be needed only in case of offset to be enabled-disabled dynamically, and should not be needed in other cases: function LL_ADC_SetOffset() automatically enables the offset.</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.</li> </ul>                                                                    |

- Reference Manual to  
LL API cross  
reference:
- OFR1 OFFSET1\_EN LL\_ADC\_SetOffsetState
  - OFR2 OFFSET2\_EN LL\_ADC\_SetOffsetState
  - OFR3 OFFSET3\_EN LL\_ADC\_SetOffsetState
  - OFR4 OFFSET4\_EN LL\_ADC\_SetOffsetState

### **LL\_ADC\_GetOffsetState**

|                                                   |                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_ADC_GetOffsetState(ADC_TypeDef * ADCx, uint32_t Offsety)</code></b>                                                                                                                                                                                                          |
| Function description                              | Get for the ADC selected offset number 1, 2, 3 or 4: offset state disabled or enabled.                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Offsety:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_OFFSET_1</li> <li>- LL_ADC_OFFSET_2</li> <li>- LL_ADC_OFFSET_3</li> <li>- LL_ADC_OFFSET_4</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_OFFSET_DISABLE</li> <li>- LL_ADC_OFFSET_ENABLE</li> </ul> </li> </ul>                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OFR1 OFFSET1_EN LL_ADC_SetOffsetState</li> <li>• OFR2 OFFSET2_EN LL_ADC_SetOffsetState</li> <li>• OFR3 OFFSET3_EN LL_ADC_SetOffsetState</li> <li>• OFR4 OFFSET4_EN LL_ADC_SetOffsetState</li> </ul>                                                                      |

### **LL\_ADC\_REG\_SetTriggerSource**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_ADC_REG_SetTriggerSource(ADC_TypeDef * ADCx, uint32_t TriggerSource)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Set ADC group regular conversion trigger source: internal (SW start) or from external IP (timer event, external interrupt line).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>TriggerSource:</b> This parameter can be one of the following values: (1) On STM32F3, parameter not available on all devices: among others, on STM32F303xE, STM32F398xx. <ul style="list-style-type: none"> <li>- LL_ADC_REG_TRIG_SOFTWARE</li> <li>- LL_ADC_REG_TRIG_EXT_TIM1_TRGO</li> <li>- LL_ADC_REG_TRIG_EXT_TIM1_TRGO2</li> <li>- LL_ADC_REG_TRIG_EXT_TIM1_CH1 (3)(4)(5)(6)</li> <li>- LL_ADC_REG_TRIG_EXT_TIM1_CH1_ADC12 (1)(2)(7)</li> <li>- LL_ADC_REG_TRIG_EXT_TIM1_CH2 (3)(4)(5)(6)</li> <li>- LL_ADC_REG_TRIG_EXT_TIM1_CH2_ADC12 (1)(2)(7)</li> <li>- LL_ADC_REG_TRIG_EXT_TIM1_CH3</li> <li>- LL_ADC_REG_TRIG_EXT_TIM2_TRGO (3)(4)(5)(6)</li> <li>- LL_ADC_REG_TRIG_EXT_TIM2_TRGO_ADC12 (1)(2)(7)</li> <li>- LL_ADC_REG_TRIG_EXT_TIM2_TRGO_ADC34 (1)(2)(8)</li> </ul> </li> </ul> |

- LL\_ADC\_REG\_TRIG\_EXT\_TIM2\_CH1\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM2\_CH2\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM2\_CH3\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_TRGO (3)(4)(5)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_TRGO\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_TRGO\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_CH1\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_CH4 (3)(4)(5)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_CH4\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM4\_TRGO (1)(2)(3)(5)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM4\_CH1\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM4\_CH4 (3) (5)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM4\_CH4\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM6\_TRGO (3)(4)(5)(6)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM6\_TRGO\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_TRGO (3)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_TRGO\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_TRGO2 (1)(2)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM6\_TRGO\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM7\_TRGO\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_TRGO\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_TRGO2 (1)(2)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_CH1\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM15\_TRGO (5)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_TRGO\_ADC12 (1)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_TRG02\_ADC12 (1)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_CH1\_ADC12 (1) (7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_CH2\_ADC12 (1) (7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_CH3\_ADC12 (1) (7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_TRG0\_ADC3 (1) (8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_TRG02\_ADC34 (1)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_CH1\_ADC34 (1) (8)
- LL\_ADC\_REG\_TRIG\_EXT\_HRTIM\_TRG1 (4)
- LL\_ADC\_REG\_TRIG\_EXT\_HRTIM\_TRG3 (4)
- LL\_ADC\_REG\_TRIG\_EXT EXTI\_LINE2\_ADC34 (1)(2)

- (8)
  - LL\_ADC\_REG\_TRIG\_EXT EXTI\_LINE11 (3)(4)(5)(6)
  - LL\_ADC\_REG\_TRIG\_EXT EXTI\_LINE11\_ADC12 (1)(2)
  - (7)
- (2) On STM32F3, parameter not available on all devices: among others, on STM32F303xC, STM32F358xx.
- (3) On STM32F3, parameter not available on all devices: among others, on STM32F303x8, STM32F328xx.
- (4) On STM32F3, parameter not available on all devices: among others, on STM32F334x8.
- (5) On STM32F3, parameter not available on all devices: among others, on STM32F302xC, STM32F302xE.
- (6) On STM32F3, parameter not available on all devices: among others, on STM32F301x8, STM32F302x8, STM32F318xx.
- (7) On STM32F3, parameter not available on all ADC instances: ADC1, ADC2 (for ADC instances ADCx available on the selected device).
- (8) On STM32F3, parameter not available on all ADC instances: ADC3, ADC4 (for ADC instances ADCx available on the selected device).

**Return values**

- **None**

**Notes**

- On this STM32 serie, setting trigger source to external trigger also set trigger polarity to rising edge (default setting for compatibility with some ADC on other STM32 families having this setting set by HW default value). In case of need to modify trigger edge, use function `LL_ADC_REG_SetTriggerEdge()`.
- Availability of parameters of trigger sources from timer depends on timers availability on the selected device.
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on group regular.

**Reference Manual to  
LL API cross  
reference:**

- CFGR EXTSEL `LL_ADC_REG_SetTriggerSource`
- CFGR EXTEN `LL_ADC_REG_SetTriggerSource`

**`LL_ADC_REG_GetTriggerSource`****Function name**

```
_STATIC_INLINE uint32_t LL_ADC_REG_GetTriggerSource
(ADC_TypeDef * ADCx)
```

**Function description**

Get ADC group regular conversion trigger source: internal (SW start) or from external IP (timer event, external interrupt line).

**Parameters**

- **ADCx:** ADC instance

**Return values**

- **Returned:** value can be one of the following values: (1) On STM32F3, parameter not available on all devices: among others, on STM32F303xE, STM32F398xx.
  - `LL_ADC_REG_TRIG_SOFTWARE`
  - `LL_ADC_REG_TRIG_EXT_TIM1_TRGO`
  - `LL_ADC_REG_TRIG_EXT_TIM1_TRGO2`
  - `LL_ADC_REG_TRIG_EXT_TIM1_CH1` (3)(4)(5)(6)

- LL\_ADC\_REG\_TRIG\_EXT\_TIM1\_CH1\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM1\_CH2 (3)(4)(5)(6)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM1\_CH2\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM1\_CH3
- LL\_ADC\_REG\_TRIG\_EXT\_TIM2\_TRGO (3)(4)(5)(6)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM2\_TRGO\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM2\_TRGO\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM2\_CH1\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM2\_CH2\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM2\_CH3\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_TRGO (3)(4)(5)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_TRGO\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_TRGO\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_CH1\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_CH4 (3)(4)(5)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM3\_CH4\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM4\_TRGO (1)(2)(3)(5)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM4\_CH1\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM4\_CH4 (3) (5)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM4\_CH4\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM6\_TRGO (3)(4)(5)(6)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM6\_TRGO\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_TRGO (3)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_TRGO\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_TRGO2 (1)(2)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM6\_TRGO\_ADC12 (1)(2)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM7\_TRGO\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_TRGO\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_TRGO2 (1)(2)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM8\_CH1\_ADC34 (1)(2)  
(8)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM15\_TRGO (5)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_TRGO\_ADC12 (1)  
(7)
- LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_TRGO2\_ADC12 (1)

- (7)
  - LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_CH1\_ADC12 (1) (7)
  - LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_CH2\_ADC12 (1) (7)
  - LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_CH3\_ADC12 (1) (7)
  - LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_TRG0\_ADC3 (1) (8)
  - LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_TRG02\_ADC34 (1) (8)
  - LL\_ADC\_REG\_TRIG\_EXT\_TIM20\_CH1\_ADC34 (1) (8)
  - LL\_ADC\_REG\_TRIG\_EXT\_HRTIM\_TRG1 (4)
  - LL\_ADC\_REG\_TRIG\_EXT\_HRTIM\_TRG3 (4)
  - LL\_ADC\_REG\_TRIG\_EXT\_EXTI\_LINE2\_ADC34 (1)(2) (8)
  - LL\_ADC\_REG\_TRIG\_EXT\_EXTI\_LINE11 (3)(4)(5)(6)
  - LL\_ADC\_REG\_TRIG\_EXT\_EXTI\_LINE11\_ADC12 (1)(2) (7)
- (2) On STM32F3, parameter not available on all devices: among others, on STM32F303xC, STM32F358xx.
- (3) On STM32F3, parameter not available on all devices: among others, on STM32F303x8, STM32F328xx.
- (4) On STM32F3, parameter not available on all devices: among others, on STM32F334x8.
- (5) On STM32F3, parameter not available on all devices: among others, on STM32F302xC, STM32F302xE.
- (6) On STM32F3, parameter not available on all devices: among others, on STM32F301x8, STM32F302x8, STM32F318xx.
- (7) On STM32F3, parameter not available on all ADC instances: ADC1, ADC2 (for ADC instances ADCx available on the selected device).
- (8) On STM32F3, parameter not available on all ADC instances: ADC3, ADC4 (for ADC instances ADCx available on the selected device).

#### Notes

- To determine whether group regular trigger source is internal (SW start) or external, without detail of which peripheral is selected as external trigger, (equivalent to "if(LL\_ADC\_REG\_GetTriggerSource(ADC1) == LL\_ADC\_REG\_TRIG\_SOFTWARE)" use function LL\_ADC\_REG\_IsTriggerSourceSWStart.
- Availability of parameters of trigger sources from timer depends on timers availability on the selected device.

#### Reference Manual to LL API cross reference:

- CFGR EXTSEL LL\_ADC\_REG\_GetTriggerSource
- CFGR EXTEN LL\_ADC\_REG\_GetTriggerSource

**LL\_ADC\_REG\_IsTriggerSourceSWStart**

|                                                   |                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_REG_IsTriggerSourceSWStart (ADC_TypeDef * ADCx)</code>                                                                                                                                    |
| Function description                              | Get ADC group regular conversion trigger source internal (SW start) or external.                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> "0" if trigger source external trigger Value "1" if trigger source SW start.</li> </ul>                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• In case of group regular trigger source set to external trigger, to determine which peripheral is selected as external trigger, use function LL_ADC_REG_GetTriggerSource().</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR EXTN LL_ADC_REG_IsTriggerSourceSWStart</li> </ul>                                                                                                                                 |

**LL\_ADC\_REG\_SetTriggerEdge**

|                                                   |                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_REG_SetTriggerEdge (ADC_TypeDef * ADCx, uint32_t ExternalTriggerEdge)</code>                                                                                                                                                                                                                              |
| Function description                              | Set ADC group regular conversion trigger polarity.                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>ExternalTriggerEdge:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_REG_TRIG_EXT_RISING</li> <li>- LL_ADC_REG_TRIG_EXT_FALLING</li> <li>- LL_ADC_REG_TRIG_EXT_RISINGFALLING</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Applicable only for trigger source set to external trigger.</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on group regular.</li> </ul>                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR EXTN LL_ADC_REG_SetTriggerEdge</li> </ul>                                                                                                                                                                                                                                                     |

**LL\_ADC\_REG\_GetTriggerEdge**

|                      |                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_REG_GetTriggerEdge (ADC_TypeDef * ADCx)</code>                                                                                                                                                                                               |
| Function description | Get ADC group regular conversion trigger polarity.                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_REG_TRIG_EXT_RISING</li> <li>- LL_ADC_REG_TRIG_EXT_FALLING</li> <li>- LL_ADC_REG_TRIG_EXT_RISINGFALLING</li> </ul> </li> </ul> |

- |                                                   |                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Notes                                             | <ul style="list-style-type: none"> <li>• Applicable only for trigger source set to external trigger.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR EXTEN LL_ADC_REG_GetTriggerEdge</li> </ul>                        |

### **LL\_ADC\_REG\_SetSequencerLength**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE void LL_ADC_REG_SetSequencerLength(ADC_TypeDef * ADCx, uint32_t SequencerNbRanks)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Set ADC group regular sequencer length and scan direction.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>SequencerNbRanks:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_REG_SEQ_SCAN_DISABLE</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_2RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_3RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_4RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_5RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_6RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_7RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_8RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_9RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_10RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_11RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_12RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_13RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_14RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_15RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_16RANKS</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• Description of ADC group regular sequencer features: For devices with sequencer fully configurable (function "LL_ADC_REG_SetSequencerRanks()" available): sequencer length and each rank affection to a channel are configurable. This function performs configuration of:<br/>Sequence length: Number of ranks in the scan sequence.Sequence direction: Unless specified in parameters, sequencer scan direction is forward (from rank 1 to rank n). Sequencer ranks are selected using function "LL_ADC_REG_SetSequencerRanks()". For devices with sequencer not fully configurable (function "LL_ADC_REG_SetSequencerChannels()" available): sequencer length and each rank affection to a channel are defined by channel number. This function performs configuration of:<br/>Sequence length: Number of ranks in the scan sequence is defined by number of channels set in the sequence, rank of each channel is fixed by channel HW number. (channel 0 fixed on rank 0, channel 1 fixed on rank1, ...).Sequence direction: Unless specified in parameters, sequencer scan direction is forward (from lowest channel number to highest channel number). Sequencer ranks are</li> </ul> |

- selected using function "LL\_ADC\_REG\_SetSequencerChannels()".
  - Sequencer disabled is equivalent to sequencer of 1 rank: ADC conversion on only 1 channel.
  - On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on group regular.
- Reference Manual to LL API cross reference:
- SQR1 L LL\_ADC\_REG\_SetSequencerLength

### **LL\_ADC\_REG\_GetSequencerLength**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_REG_GetSequencerLength (ADC_TypeDef * ADCx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Get ADC group regular sequencer length and scan direction.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_REG_SEQ_SCAN_DISABLE</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_2RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_3RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_4RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_5RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_6RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_7RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_8RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_9RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_10RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_11RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_12RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_13RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_14RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_15RANKS</li> <li>- LL_ADC_REG_SEQ_SCAN_ENABLE_16RANKS</li> </ul> </li> </ul>                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• Description of ADC group regular sequencer features: For devices with sequencer fully configurable (function "LL_ADC_REG_SetSequencerRanks()" available): sequencer length and each rank affectation to a channel are configurable. This function retrieves: Sequence length: Number of ranks in the scan sequence. Sequence direction: Unless specified in parameters, sequencer scan direction is forward (from rank 1 to rank n). Sequencer ranks are selected using function "LL_ADC_REG_SetSequencerRanks()". For devices with sequencer not fully configurable (function "LL_ADC_REG_SetSequencerChannels()" available): sequencer length and each rank affectation to a channel are defined by channel number. This function retrieves: Sequence length: Number of ranks in the scan sequence is defined by number of channels set in the sequence, rank of each channel is fixed by channel HW number. (channel 0 fixed on rank 0, channel 1 fixed on rank1, ...). Sequence direction:</li> </ul> |

Reference Manual to  
LL API cross  
reference:

Unless specified in parameters, sequencer scan direction is forward (from lowest channel number to highest channel number). Sequencer ranks are selected using function "LL\_ADC\_REG\_SetSequencerChannels()".

- Sequencer disabled is equivalent to sequencer of 1 rank: ADC conversion on only 1 channel.
- SQR1 L LL\_ADC\_REG\_GetSequencerLength

### **LL\_ADC\_REG\_SetSequencerDiscont**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_ADC_REG_SetSequencerDiscont(ADC_TypeDef * ADCx, uint32_t SeqDiscont)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Set ADC group regular sequencer discontinuous mode: sequence subdivided and scan conversions interrupted every selected number of ranks.                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>SeqDiscont:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_REG_SEQ_DISCONT_DISABLE</li> <li>- LL_ADC_REG_SEQ_DISCONT_1RANK</li> <li>- LL_ADC_REG_SEQ_DISCONT_2RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_3RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_4RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_5RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_6RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_7RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_8RANKS</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• It is not possible to enable both ADC group regular continuous mode and sequencer discontinuous mode.</li> <li>• It is not possible to enable both ADC auto-injected mode and ADC group regular sequencer discontinuous mode.</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on group regular.</li> </ul>                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR DISCEN LL_ADC_REG_SetSequencerDiscont</li> <li>• CFGR DISCNUM LL_ADC_REG_SetSequencerDiscont</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                     |

### **LL\_ADC\_REG\_GetSequencerDiscont**

|                      |                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_ADC_REG_GetSequencerDiscont(ADC_TypeDef * ADCx)</code></b>                                          |
| Function description | Get ADC group regular sequencer discontinuous mode: sequence subdivided and scan conversions interrupted every selected number of ranks. |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                            |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_ADC_REG_SEQ_DISCONT_DISABLE</li> <li>- LL_ADC_REG_SEQ_DISCONT_1RANK</li> <li>- LL_ADC_REG_SEQ_DISCONT_2RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_3RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_4RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_5RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_6RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_7RANKS</li> <li>- LL_ADC_REG_SEQ_DISCONT_8RANKS</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR DISCEN LL_ADC_REG_GetSequencerDiscont</li> <li>• CFGR DISCNUM LL_ADC_REG_GetSequencerDiscont</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                               |

## LL\_ADC\_REG\_SetSequencerRanks

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_REG_SetSequencerRanks(ADC_TypeDef * ADCx, uint32_t Rank, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Set ADC group regular sequence: channel on the selected scan sequence rank.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Rank:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_ADC_REG_RANK_1</li> <li>- LL_ADC_REG_RANK_2</li> <li>- LL_ADC_REG_RANK_3</li> <li>- LL_ADC_REG_RANK_4</li> <li>- LL_ADC_REG_RANK_5</li> <li>- LL_ADC_REG_RANK_6</li> <li>- LL_ADC_REG_RANK_7</li> <li>- LL_ADC_REG_RANK_8</li> <li>- LL_ADC_REG_RANK_9</li> <li>- LL_ADC_REG_RANK_10</li> <li>- LL_ADC_REG_RANK_11</li> <li>- LL_ADC_REG_RANK_12</li> <li>- LL_ADC_REG_RANK_13</li> <li>- LL_ADC_REG_RANK_14</li> <li>- LL_ADC_REG_RANK_15</li> <li>- LL_ADC_REG_RANK_16</li> </ul> </li> <li>• <b>Channel:</b> This parameter can be one of the following values:<br/>(1) On STM32F3, parameter available only on ADC instance: ADC1.           <ul style="list-style-type: none"> <li>- LL_ADC_CHANNEL_0</li> <li>- LL_ADC_CHANNEL_1</li> <li>- LL_ADC_CHANNEL_2</li> <li>- LL_ADC_CHANNEL_3</li> <li>- LL_ADC_CHANNEL_4</li> <li>- LL_ADC_CHANNEL_5</li> <li>- LL_ADC_CHANNEL_6</li> <li>- LL_ADC_CHANNEL_7</li> <li>- LL_ADC_CHANNEL_8</li> <li>- LL_ADC_CHANNEL_9</li> </ul> </li> </ul> |

- LL\_ADC\_CHANNEL\_10
  - LL\_ADC\_CHANNEL\_11
  - LL\_ADC\_CHANNEL\_12
  - LL\_ADC\_CHANNEL\_13
  - LL\_ADC\_CHANNEL\_14
  - LL\_ADC\_CHANNEL\_15
  - LL\_ADC\_CHANNEL\_16
  - LL\_ADC\_CHANNEL\_17
  - LL\_ADC\_CHANNEL\_18
  - LL\_ADC\_CHANNEL\_VREFINT (5)
  - LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
  - LL\_ADC\_CHANNEL\_VBAT (1)
  - LL\_ADC\_CHANNEL\_VOPAMP1 (1)
  - LL\_ADC\_CHANNEL\_VOPAMP2 (2)
  - LL\_ADC\_CHANNEL\_VOPAMP3 (3)
  - LL\_ADC\_CHANNEL\_VOPAMP4 (4)
  - (2) On STM32F3, parameter available only on ADC instance: ADC2.
  - (3) On STM32F3, parameter available only on ADC instance: ADC3.
  - (4) On STM32F3, parameter available only on ADC instances: ADC4.
  - (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.
- Return values**
- **None**
- Notes**
- This function performs configuration of: Channels ordering into each rank of scan sequence: whatever channel can be placed into whatever rank.
  - On this STM32 serie, ADC group regular sequencer is fully configurable: sequencer length and each rank affection to a channel are configurable. Refer to description of function LL\_ADC\_REG\_SetSequencerLength().
  - Depending on devices and packages, some channels may not be available. Refer to device datasheet for channels availability.
  - On this STM32 serie, to measure internal channels (VrefInt, TempSensor, ...), measurement paths to internal channels must be enabled separately. This can be done using function LL\_ADC\_SetCommonPathInternalCh().
  - On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on group regular.
- Reference Manual to LL API cross reference:**
- SQR1 SQ1 LL\_ADC\_REG\_SetSequencerRanks
  - SQR1 SQ2 LL\_ADC\_REG\_SetSequencerRanks
  - SQR1 SQ3 LL\_ADC\_REG\_SetSequencerRanks
  - SQR1 SQ4 LL\_ADC\_REG\_SetSequencerRanks
  - SQR2 SQ5 LL\_ADC\_REG\_SetSequencerRanks
  - SQR2 SQ6 LL\_ADC\_REG\_SetSequencerRanks
  - SQR2 SQ7 LL\_ADC\_REG\_SetSequencerRanks
  - SQR2 SQ8 LL\_ADC\_REG\_SetSequencerRanks
  - SQR2 SQ9 LL\_ADC\_REG\_SetSequencerRanks

- SQR3 SQ10 LL\_ADC\_REG\_SetSequencerRanks
- SQR3 SQ11 LL\_ADC\_REG\_SetSequencerRanks
- SQR3 SQ12 LL\_ADC\_REG\_SetSequencerRanks
- SQR3 SQ13 LL\_ADC\_REG\_SetSequencerRanks
- SQR3 SQ14 LL\_ADC\_REG\_SetSequencerRanks
- SQR4 SQ15 LL\_ADC\_REG\_SetSequencerRanks
- SQR4 SQ16 LL\_ADC\_REG\_SetSequencerRanks

### **LL\_ADC\_REG\_GetSequencerRanks**

Function name **`__STATIC_INLINE uint32_t LL_ADC_REG_GetSequencerRanks (ADC_TypeDef * ADCx, uint32_t Rank)`**

Function description Get ADC group regular sequence: channel on the selected scan sequence rank.

Parameters

- **ADCx:** ADC instance
- **Rank:** This parameter can be one of the following values:
  - LL\_ADC\_REG\_RANK\_1
  - LL\_ADC\_REG\_RANK\_2
  - LL\_ADC\_REG\_RANK\_3
  - LL\_ADC\_REG\_RANK\_4
  - LL\_ADC\_REG\_RANK\_5
  - LL\_ADC\_REG\_RANK\_6
  - LL\_ADC\_REG\_RANK\_7
  - LL\_ADC\_REG\_RANK\_8
  - LL\_ADC\_REG\_RANK\_9
  - LL\_ADC\_REG\_RANK\_10
  - LL\_ADC\_REG\_RANK\_11
  - LL\_ADC\_REG\_RANK\_12
  - LL\_ADC\_REG\_RANK\_13
  - LL\_ADC\_REG\_RANK\_14
  - LL\_ADC\_REG\_RANK\_15
  - LL\_ADC\_REG\_RANK\_16

Return values

- **Returned:** value can be one of the following values: (1) On STM32F3, parameter available only on ADC instance: ADC1.
  - LL\_ADC\_CHANNEL\_0
  - LL\_ADC\_CHANNEL\_1
  - LL\_ADC\_CHANNEL\_2
  - LL\_ADC\_CHANNEL\_3
  - LL\_ADC\_CHANNEL\_4
  - LL\_ADC\_CHANNEL\_5
  - LL\_ADC\_CHANNEL\_6
  - LL\_ADC\_CHANNEL\_7
  - LL\_ADC\_CHANNEL\_8
  - LL\_ADC\_CHANNEL\_9
  - LL\_ADC\_CHANNEL\_10
  - LL\_ADC\_CHANNEL\_11
  - LL\_ADC\_CHANNEL\_12
  - LL\_ADC\_CHANNEL\_13
  - LL\_ADC\_CHANNEL\_14
  - LL\_ADC\_CHANNEL\_15

- LL\_ADC\_CHANNEL\_16
  - LL\_ADC\_CHANNEL\_17
  - LL\_ADC\_CHANNEL\_18
  - LL\_ADC\_CHANNEL\_VREFINT (5)
  - LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
  - LL\_ADC\_CHANNEL\_VBAT (1)
  - LL\_ADC\_CHANNEL\_VOPAMP1 (1)
  - LL\_ADC\_CHANNEL\_VOPAMP2 (2)
  - LL\_ADC\_CHANNEL\_VOPAMP3 (3)
  - LL\_ADC\_CHANNEL\_VOPAMP4 (4)
  - (2) On STM32F3, parameter available only on ADC instance: ADC2.
  - (3) On STM32F3, parameter available only on ADC instance: ADC3.
  - (4) On STM32F3, parameter available only on ADC instances: ADC4.
  - (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.
  - (1, 2, 3, 4, 5) For ADC channel read back from ADC register, comparison with internal channel parameter to be done using helper macro  
  \_\_LL\_ADC\_CHANNEL\_INTERNAL\_TO\_EXTERNAL().
- Notes**
- On this STM32 serie, ADC group regular sequencer is fully configurable: sequencer length and each rank affectation to a channel are configurable. Refer to description of function LL\_ADC\_REG\_SetSequencerLength().
  - Depending on devices and packages, some channels may not be available. Refer to device datasheet for channels availability.
  - Usage of the returned channel number: To reinject this channel into another function LL\_ADC\_xxx: the returned channel number is only partly formatted on definition of literals LL\_ADC\_CHANNEL\_x. Therefore, it has to be compared with parts of literals LL\_ADC\_CHANNEL\_x or using helper macro \_\_LL\_ADC\_CHANNEL\_TO\_DECIMAL\_NB(). Then the selected literal LL\_ADC\_CHANNEL\_x can be used as parameter for another function. To get the channel number in decimal format: process the returned value with the helper macro \_\_LL\_ADC\_CHANNEL\_TO\_DECIMAL\_NB().
- Reference Manual to LL API cross reference:**
- SQR1 SQ1 LL\_ADC\_REG\_GetSequencerRanks
  - SQR1 SQ2 LL\_ADC\_REG\_GetSequencerRanks
  - SQR1 SQ3 LL\_ADC\_REG\_GetSequencerRanks
  - SQR1 SQ4 LL\_ADC\_REG\_GetSequencerRanks
  - SQR2 SQ5 LL\_ADC\_REG\_GetSequencerRanks
  - SQR2 SQ6 LL\_ADC\_REG\_GetSequencerRanks
  - SQR2 SQ7 LL\_ADC\_REG\_GetSequencerRanks
  - SQR2 SQ8 LL\_ADC\_REG\_GetSequencerRanks
  - SQR2 SQ9 LL\_ADC\_REG\_GetSequencerRanks
  - SQR3 SQ10 LL\_ADC\_REG\_GetSequencerRanks
  - SQR3 SQ11 LL\_ADC\_REG\_GetSequencerRanks
  - SQR3 SQ12 LL\_ADC\_REG\_GetSequencerRanks
  - SQR3 SQ13 LL\_ADC\_REG\_GetSequencerRanks

- SQR3 SQ14 LL\_ADC\_REG\_GetSequencerRanks
- SQR4 SQ15 LL\_ADC\_REG\_GetSequencerRanks
- SQR4 SQ16 LL\_ADC\_REG\_GetSequencerRanks

### **LL\_ADC\_REG\_SetContinuousMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_ADC_REG_SetContinuousMode(ADC_TypeDef * ADCx, uint32_t Continuous)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                              | Set ADC continuous conversion mode on ADC group regular.                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Continuous:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_REG_CONV_SINGLE</li> <li>- LL_ADC_REG_CONV_CONTINUOUS</li> </ul> </li> </ul>                                                                                                                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Description of ADC continuous conversion mode: single mode: one conversion per triggercontinuous mode: after the first trigger, following conversions launched successively automatically.</li> <li>• It is not possible to enable both ADC group regular continuous mode and sequencer discontinuous mode.</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on group regular.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR CONT LL_ADC_REG_SetContinuousMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                         |

### **LL\_ADC\_REG\_GetContinuousMode**

|                                                   |                                                                                                                                                                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_ADC_REG_GetContinuousMode(ADC_TypeDef * ADCx)</code></b>                                                                                                                                                   |
| Function description                              | Get ADC continuous conversion mode on ADC group regular.                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_REG_CONV_SINGLE</li> <li>- LL_ADC_REG_CONV_CONTINUOUS</li> </ul> </li> </ul>               |
| Notes                                             | <ul style="list-style-type: none"> <li>• Description of ADC continuous conversion mode: single mode: one conversion per triggercontinuous mode: after the first trigger, following conversions launched successively automatically.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR CONT LL_ADC_REG_GetContinuousMode</li> </ul>                                                                                                                                                     |

**LL\_ADC\_REG\_SetDMATransfer**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_REG_SetDMATransfer(ADC_TypeDef * ADCx, uint32_t DMATransfer)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Set ADC group regular conversion data transfer: no transfer or transfer by DMA, and DMA requests mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>DMATransfer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_ADC_REG_DMA_TRANSFER_NONE</li> <li>– LL_ADC_REG_DMA_TRANSFER_LIMITED</li> <li>– LL_ADC_REG_DMA_TRANSFER_UNLIMITED</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• If transfer by DMA selected, specifies the DMA requests mode: Limited mode (One shot mode): DMA transfer requests are stopped when number of DMA data transfers (number of ADC conversions) is reached. This ADC mode is intended to be used with DMA mode non-circular.Unlimited mode: DMA transfer requests are unlimited, whatever number of DMA data transfers (number of ADC conversions). This ADC mode is intended to be used with DMA mode circular.</li> <li>• If ADC DMA requests mode is set to unlimited and DMA is set to mode non-circular: when DMA transfers size will be reached, DMA will stop transfers of ADC conversions data ADC will raise an overrun error (overrun flag and interruption if enabled).</li> <li>• For devices with several ADC instances: ADC multimode DMA settings are available using function <code>LL_ADC_SetMultiDMATransfer()</code>.</li> <li>• To configure DMA source address (peripheral address), use function <code>LL_ADC_DMA_GetRegAddr()</code>.</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR DMAEN LL_ADC_REG_SetDMATransfer</li> <li>• CFGR DMACFG LL_ADC_REG_SetDMATransfer</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

**LL\_ADC\_REG\_GetDMATransfer**

|                      |                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_REG_GetDMATransfer(ADC_TypeDef * ADCx)</code>                                                                                                                                                                                                      |
| Function description | Get ADC group regular conversion data transfer: no transfer or transfer by DMA, and DMA requests mode.                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                                            |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_ADC_REG_DMA_TRANSFER_NONE</li> <li>– LL_ADC_REG_DMA_TRANSFER_LIMITED</li> <li>– LL_ADC_REG_DMA_TRANSFER_UNLIMITED</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• If transfer by DMA selected, specifies the DMA requests</li> </ul>                                                                                                                                                                              |

mode: Limited mode (One shot mode): DMA transfer requests are stopped when number of DMA data transfers (number of ADC conversions) is reached. This ADC mode is intended to be used with DMA mode non-circular. Unlimited mode: DMA transfer requests are unlimited, whatever number of DMA data transfers (number of ADC conversions). This ADC mode is intended to be used with DMA mode circular.

- If ADC DMA requests mode is set to unlimited and DMA is set to mode non-circular: when DMA transfers size will be reached, DMA will stop transfers of ADC conversions data ADC will raise an overrun error (overrun flag and interruption if enabled).
- For devices with several ADC instances: ADC multimode DMA settings are available using function LL\_ADC\_GetMultiDMATransfer().
- To configure DMA source address (peripheral address), use function LL\_ADC\_DMA\_GetRegAddr().

Reference Manual to  
LL API cross  
reference:

- CFGR DMAEN LL\_ADC\_REG\_GetDMATransfer
- CFGR DMACFG LL\_ADC\_REG\_GetDMATransfer

### LL\_ADC\_REG\_SetOverrun

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_ADC_REG_SetOverrun(ADC_TypeDef * ADCx, uint32_t Overrun)</code>                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description                              | Set ADC group regular behavior in case of overrun: data preserved or overwritten.                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Overrun:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_REG_OVR_DATA_PRESERVED</li> <li>- LL_ADC_REG_OVR_DATA_OVERWRITTEN</li> </ul> </li> </ul>                                                                                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• Compatibility with devices without feature overrun: other devices without this feature have a behavior equivalent to data overwritten. The default setting of overrun is data preserved. Therefore, for compatibility with all devices, parameter overrun should be set to data overwritten.</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on group regular.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR OVRMOD LL_ADC_REG_SetOverrun</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                               |

### LL\_ADC\_REG\_GetOverrun

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_ADC_REG_GetOverrun(ADC_TypeDef * ADCx)</code>    |
| Function description | Get ADC group regular behavior in case of overrun: data preserved or overwritten. |

- |                                                   |                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | • <b>ADCx:</b> ADC instance                                                                                                          |
| Return values                                     | • <b>Returned:</b> value can be one of the following values:<br>– LL_ADC_REG_OVR_DATA_PRESERVED<br>– LL_ADC_REG_OVR_DATA_OVERWRITTEN |
| Reference Manual to<br>LL API cross<br>reference: | • CFGR OVRMOD LL_ADC_REG_GetOverrun                                                                                                  |

### **LL\_ADC\_INJ\_SetTriggerSource**

- |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>__STATIC_INLINE void LL_ADC_INJ_SetTriggerSource(ADC_TypeDef * ADCx, uint32_t TriggerSource)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Set ADC group injected conversion trigger source: internal (SW start) or from external IP (timer event, external interrupt line).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>TriggerSource:</b> This parameter can be one of the following values: (1) On STM32F3, parameter not available on all devices: among others, on STM32F303xE, STM32F398xx.           <ul style="list-style-type: none"> <li>– LL_ADC_INJ_TRIG_SOFTWARE</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM1_TRGO</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM1_TRGO2</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM1_CH3_ADC34 (1)(2) (8)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM1_CH4</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM2_TRGO (3)(4)(5)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM2_TRGO_ADC12 (1)(2) (7)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM2_TRGO_ADC34 (1)(2) (8)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM2_CH1 (3)(4)(5)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM2_CH1_ADC12 (1)(2) (7)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM3_TRGO (1)(2)(3)(4)(5)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM3_CH1 (3)(4)(5)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM3_CH1_ADC12 (1)(2) (7)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM3_CH3 (3)(4)(5)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM3_CH3_ADC12 (1)(2) (7)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM3_CH4 (3)(4)(5)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM3_CH4_ADC12 (1)(2)(3)(4)(5) (7)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM4_TRGO (3) (5)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM4_TRGO_ADC12 (1)(2) (7)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM4_TRGO_ADC34 (1)(2) (8)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM4_CH3_ADC34 (1)(2) (8)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM4_CH4_ADC34 (1)(2) (8)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM6_TRGO (3)(4)(5)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM6_TRGO_ADC12 (1)(2) (7)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM7_TRGO_ADC34 (1)(2) (8)</li> <li>– LL_ADC_INJ_TRIG_EXT_TIM8_TRGO (1)(2)</li> </ul> </li> </ul> |

- LL\_ADC\_INJ\_TRIG\_EXT\_TIM8\_TRGO2 (1)(2)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM8\_CH2\_ADC34 (1)(2) (8)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM8\_CH4\_ADC12 (1)(2) (7)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM8\_CH4\_ADC34 (1)(2) (8)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM15\_TRGO
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRGO\_ADC12 (1) (7)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRGO2\_ADC12 (1) (7)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_CH4\_ADC12 (1) (7)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRG\_ADC34 (1) (8)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRG2\_ADC34 (1) (8)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_CH2\_ADC34 (1) (8)
- LL\_ADC\_INJ\_TRIG\_EXT\_HRTIM\_TRG2 (4)
- LL\_ADC\_INJ\_TRIG\_EXT\_HRTIM\_TRG4 (4)
- LL\_ADC\_INJ\_TRIG\_EXT\_EXTI\_LINE15 (3)(4)(5)(6)
- LL\_ADC\_INJ\_TRIG\_EXT\_EXTI\_LINE15\_ADC12 (1)(2) (7)
- (2) On STM32F3, parameter not available on all devices: among others, on STM32F303xC, STM32F358xx.
- (3) On STM32F3, parameter not available on all devices: among others, on STM32F303x8, STM32F328xx.
- (4) On STM32F3, parameter not available on all devices: among others, on STM32F334x8.
- (5) On STM32F3, parameter not available on all devices: among others, on STM32F302xC, STM32F302xE.
- (6) On STM32F3, parameter not available on all devices: among others, on STM32F301x8, STM32F302x8, STM32F318xx.
- (7) On STM32F3, parameter not available on all ADC instances: ADC1, ADC2 (for ADC instances ADCx available on the selected device).
- (8) On STM32F3, parameter not available on all ADC instances: ADC3, ADC4 (for ADC instances ADCx available on the selected device).

#### Return values

- **None**

#### Notes

- On this STM32 serie, setting trigger source to external trigger also set trigger polarity to rising edge (default setting for compatibility with some ADC on other STM32 families having this setting set by HW default value). In case of need to modify trigger edge, use function LL\_ADC\_INJ\_SetTriggerEdge().
- Caution to ADC group injected contexts queue: On this STM32 serie, using successively several times this function will appear has having no effect. This is due to ADC group injected contexts queue (this feature cannot be disabled on this STM32 serie). To set several features of ADC group injected, use function LL\_ADC\_INJ\_ConfigQueueContext().
- Availability of parameters of trigger sources from timer depends on timers availability on the selected device.
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must not be disabled. Can be enabled with or without conversion on going on either groups regular or

injected.

Reference Manual to  
LL API cross  
reference:

- JSQR JEXTSEL LL\_ADC\_INJ\_SetTriggerSource
- JSQR JEXTEN LL\_ADC\_INJ\_SetTriggerSource

## LL\_ADC\_INJ\_GetTriggerSource

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_INJ_GetTriggerSource(ADC_TypeDef * ADCx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Get ADC group injected conversion trigger source: internal (SW start) or from external IP (timer event, external interrupt line).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Return values        | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values: (1) On STM32F3, parameter not available on all devices: among others, on STM32F303xE, STM32F398xx. <ul style="list-style-type: none"> <li>- LL_ADC_INJ_TRIG_SOFTWARE</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM1_TRGO</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM1_TRGO2</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM1_CH3_ADC34 (1)(2) (8)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM1_CH4</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM2_TRGO (3)(4)(5)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM2_TRGO_ADC12 (1)(2) (7)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM2_TRGO_ADC34 (1)(2) (8)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM2_CH1 (3)(4)(5)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM2_CH1_ADC12 (1)(2) (7)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM3_TRGO (1)(2)(3)(4)(5)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM3_CH1 (3)(4)(5)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM3_CH1_ADC12 (1)(2) (7)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM3_CH3 (3)(4)(5)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM3_CH3_ADC12 (1)(2) (7)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM3_CH4 (3)(4)(5)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM3_CH4_ADC12 (1)(2)(3)(4)(5) (7)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM4_TRGO (3) (5)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM4_TRGO_ADC12 (1)(2) (7)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM4_TRGO_ADC34 (1)(2) (8)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM4_CH3_ADC34 (1)(2) (8)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM4_CH4_ADC34 (1)(2) (8)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM6_TRGO (3)(4)(5)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM6_TRGO_ADC12 (1)(2) (7)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM7_TRGO_ADC34 (1)(2) (8)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM8_TRGO (1)(2)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM8_TRGO2 (1)(2)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM8_CH2_ADC34 (1)(2) (8)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM8_CH4_ADC12 (1)(2) (7)</li> </ul> </li> </ul> |

- LL\_ADC\_INJ\_TRIG\_EXT\_TIM8\_CH4\_ADC34 (1)(2) (8)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM15\_TRGO
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRGO\_ADC12 (1) (7)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRGO2\_ADC12 (1) (7)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_CH4\_ADC12 (1) (7)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRG\_ADC34 (1) (8)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRG2\_ADC34 (1) (8)
- LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_CH2\_ADC34 (1) (8)
- LL\_ADC\_INJ\_TRIG\_EXT\_HRTIM\_TRG2 (4)
- LL\_ADC\_INJ\_TRIG\_EXT\_HRTIM\_TRG4 (4)
- LL\_ADC\_INJ\_TRIG\_EXT\_EXTI\_LINE15 (3)(4)(5)(6)
- LL\_ADC\_INJ\_TRIG\_EXT\_EXTI\_LINE15\_ADC12 (1)(2) (7)
- (2) On STM32F3, parameter not available on all devices: among others, on STM32F303xC, STM32F358xx.
- (3) On STM32F3, parameter not available on all devices: among others, on STM32F303x8, STM32F328xx.
- (4) On STM32F3, parameter not available on all devices: among others, on STM32F334x8.
- (5) On STM32F3, parameter not available on all devices: among others, on STM32F302xC, STM32F302xE.
- (6) On STM32F3, parameter not available on all devices: among others, on STM32F301x8, STM32F302x8, STM32F318xx.
- (7) On STM32F3, parameter not available on all ADC instances: ADC1, ADC2 (for ADC instances ADCx available on the selected device).
- (8) On STM32F3, parameter not available on all ADC instances: ADC3, ADC4 (for ADC instances ADCx available on the selected device).

#### Notes

- To determine whether group injected trigger source is internal (SW start) or external, without detail of which peripheral is selected as external trigger, (equivalent to "if(LL\_ADC\_INJ\_GetTriggerSource(ADC1) == LL\_ADC\_INJ\_TRIG\_SOFTWARE)" use function LL\_ADC\_INJ\_IsTriggerSourceSWStart.
- Availability of parameters of trigger sources from timer depends on timers availability on the selected device.

#### Reference Manual to LL API cross reference:

- JSQR JEXTSEL LL\_ADC\_INJ\_GetTriggerSource
- JSQR JEXTEN LL\_ADC\_INJ\_GetTriggerSource

### **LL\_ADC\_INJ\_IsTriggerSourceSWStart**

|                      |                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>STATIC_INLINE uint32_t<br/>LL_ADC_INJ_IsTriggerSourceSWStart (ADC_TypeDef * ADCx)</b>                                      |
| Function description | Get ADC group injected conversion trigger source internal (SW start) or external.                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                 |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Value:</b> "0" if trigger source external trigger Value "1" if trigger</li> </ul> |

- source SW start.
- |                                                   |                                                                                                                                                                                                                              |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                             | <ul style="list-style-type: none"> <li>In case of group injected trigger source set to external trigger, to determine which peripheral is selected as external trigger, use function LL_ADC_INJ_GetTriggerSource.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>JSQR JEXTEN LL_ADC_INJ_IsTriggerSourceSWStart</li> </ul>                                                                                                                              |

### **LL\_ADC\_INJ\_SetTriggerEdge**

- |                                                   |                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_ADC_INJ_SetTriggerEdge(ADC_TypeDef * ADCx, uint32_t ExternalTriggerEdge)</code></b>                                                                                                                                                                                                                               |
| Function description                              | Set ADC group injected conversion trigger polarity.                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> <li><b>ExternalTriggerEdge:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_ADC_INJ_TRIG_EXT_RISING</li> <li>– LL_ADC_INJ_TRIG_EXT_FALLING</li> <li>– LL_ADC_INJ_TRIG_EXT_RISINGFALLING</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must not be disabled. Can be enabled with or without conversion on going on either groups regular or injected.</li> </ul>                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>JSQR JEXTEN LL_ADC_INJ_SetTriggerEdge</li> </ul>                                                                                                                                                                                                                                                           |

### **LL\_ADC\_INJ\_GetTriggerEdge**

- |                                                   |                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_ADC_INJ_GetTriggerEdge(ADC_TypeDef * ADCx)</code></b>                                                                                                                                                                                                  |
| Function description                              | Get ADC group injected conversion trigger polarity.                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_ADC_INJ_TRIG_EXT_RISING</li> <li>– LL_ADC_INJ_TRIG_EXT_FALLING</li> <li>– LL_ADC_INJ_TRIG_EXT_RISINGFALLING</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>JSQR JEXTEN LL_ADC_INJ_SetTriggerEdge</li> </ul>                                                                                                                                                                                                    |

**LL\_ADC\_INJ\_SetSequencerLength**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_INJ_SetSequencerLength(ADC_TypeDef * ADCx, uint32_t SequencerNbRanks)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                              | Set ADC group injected sequencer length and scan direction.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>SequencerNbRanks:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_INJ_SEQ_SCAN_DISABLE</li> <li>- LL_ADC_INJ_SEQ_SCAN_ENABLE_2RANKS</li> <li>- LL_ADC_INJ_SEQ_SCAN_ENABLE_3RANKS</li> <li>- LL_ADC_INJ_SEQ_SCAN_ENABLE_4RANKS</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function performs configuration of: Sequence length: Number of ranks in the scan sequence. Sequence direction: Unless specified in parameters, sequencer scan direction is forward (from rank 1 to rank n).</li> <li>• Sequencer disabled is equivalent to sequencer of 1 rank: ADC conversion on only 1 channel.</li> <li>• Caution to ADC group injected contexts queue: On this STM32 serie, using successively several times this function will appear has having no effect. This is due to ADC group injected contexts queue (this feature cannot be disabled on this STM32 serie). To set several features of ADC group injected, use function <code>LL_ADC_INJ_ConfigQueueContext()</code>.</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must not be disabled. Can be enabled with or without conversion on going on either groups regular or injected.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• JSQR JL <code>LL_ADC_INJ_SetSequencerLength</code></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

**LL\_ADC\_INJ\_GetSequencerLength**

|                      |                                                                                                                                                                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_INJ_GetSequencerLength(ADC_TypeDef * ADCx)</code>                                                                                                                                                                                                                                                |
| Function description | Get ADC group injected sequencer length and scan direction.                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_INJ_SEQ_SCAN_DISABLE</li> <li>- LL_ADC_INJ_SEQ_SCAN_ENABLE_2RANKS</li> <li>- LL_ADC_INJ_SEQ_SCAN_ENABLE_3RANKS</li> <li>- LL_ADC_INJ_SEQ_SCAN_ENABLE_4RANKS</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• This function retrieves: Sequence length: Number of ranks in the scan sequence. Sequence direction: Unless specified in parameters, sequencer scan direction is forward (from rank 1 to rank n).</li> <li>• Sequencer disabled is equivalent to sequencer of 1 rank:</li> </ul>               |

- ADC conversion on only 1 channel.
- Reference Manual to LL API cross reference:
- JSQR JL LL\_ADC\_INJ\_GetSequencerLength

### **LL\_ADC\_INJ\_SetSequencerDiscont**

|                                             |                                                                                                                                                                                                                                                                                    |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_ADC_INJ_SetSequencerDiscont(ADC_TypeDef * ADCx, uint32_t SeqDiscont)</code></b>                                                                                                                                                                    |
| Function description                        | Set ADC group injected sequencer discontinuous mode: sequence subdivided and scan conversions interrupted every selected number of ranks.                                                                                                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> <li><b>SeqDiscont:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_ADC_INJ_SEQ_DISCONT_DISABLE</li> <li>LL_ADC_INJ_SEQ_DISCONT_1RANK</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                      |
| Notes                                       | <ul style="list-style-type: none"> <li>It is not possible to enable both ADC group injected auto-injected mode and sequencer discontinuous mode.</li> </ul>                                                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CFGJ JDISCEN LL_ADC_INJ_SetSequencerDiscont</li> </ul>                                                                                                                                                                                      |

### **LL\_ADC\_INJ\_GetSequencerDiscont**

|                                             |                                                                                                                                                                                                                                      |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_ADC_INJ_GetSequencerDiscont(ADC_TypeDef * ADCx)</code></b>                                                                                                                                       |
| Function description                        | Get ADC group injected sequencer discontinuous mode: sequence subdivided and scan conversions interrupted every selected number of ranks.                                                                                            |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                          |
| Return values                               | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>LL_ADC_INJ_SEQ_DISCONT_DISABLE</li> <li>LL_ADC_INJ_SEQ_DISCONT_1RANK</li> </ul> </li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CFGJ JDISCEN LL_ADC_INJ_SetSequencerDiscont</li> </ul>                                                                                                                                        |

### **LL\_ADC\_INJ\_SetSequencerRanks**

|                      |                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_ADC_INJ_SetSequencerRanks(ADC_TypeDef * ADCx, uint32_t Rank, uint32_t Channel)</code></b>                                                                                                 |
| Function description | Set ADC group injected sequence: channel on the selected sequence rank.                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> <li><b>Rank:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_ADC_INJ_RANK_1</li> </ul> </li> </ul> |

- LL\_ADC\_INJ\_RANK\_2
- LL\_ADC\_INJ\_RANK\_3
- LL\_ADC\_INJ\_RANK\_4
- **Channel:** This parameter can be one of the following values:
  - (1) On STM32F3, parameter available only on ADC instance: ADC1.
    - LL\_ADC\_CHANNEL\_0
    - LL\_ADC\_CHANNEL\_1
    - LL\_ADC\_CHANNEL\_2
    - LL\_ADC\_CHANNEL\_3
    - LL\_ADC\_CHANNEL\_4
    - LL\_ADC\_CHANNEL\_5
    - LL\_ADC\_CHANNEL\_6
    - LL\_ADC\_CHANNEL\_7
    - LL\_ADC\_CHANNEL\_8
    - LL\_ADC\_CHANNEL\_9
    - LL\_ADC\_CHANNEL\_10
    - LL\_ADC\_CHANNEL\_11
    - LL\_ADC\_CHANNEL\_12
    - LL\_ADC\_CHANNEL\_13
    - LL\_ADC\_CHANNEL\_14
    - LL\_ADC\_CHANNEL\_15
    - LL\_ADC\_CHANNEL\_16
    - LL\_ADC\_CHANNEL\_17
    - LL\_ADC\_CHANNEL\_18
    - LL\_ADC\_CHANNEL\_VREFINT (5)
    - LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
    - LL\_ADC\_CHANNEL\_VBAT (1)
    - LL\_ADC\_CHANNEL\_VOPAMP1 (1)
    - LL\_ADC\_CHANNEL\_VOPAMP2 (2)
    - LL\_ADC\_CHANNEL\_VOPAMP3 (3)
    - LL\_ADC\_CHANNEL\_VOPAMP4 (4)
  - (2) On STM32F3, parameter available only on ADC instance: ADC2.
  - (3) On STM32F3, parameter available only on ADC instance: ADC3.
  - (4) On STM32F3, parameter available only on ADC instances: ADC4.
  - (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.

**Return values****Notes**

- **None**
- Depending on devices and packages, some channels may not be available. Refer to device datasheet for channels availability.
- On this STM32 serie, to measure internal channels (VrefInt, TempSensor, ...), measurement paths to internal channels must be enabled separately. This can be done using function LL\_ADC\_SetCommonPathInternalCh().
- Caution to ADC group injected contexts queue: On this STM32 serie, using successively several times this function will appear has having no effect. This is due to ADC group

- injected contexts queue (this feature cannot be disabled on this STM32 serie). To set several features of ADC group injected, use function LL\_ADC\_INJ\_ConfigQueueContext().
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must not be disabled. Can be enabled with or without conversion on going on either groups regular or injected.
- Reference Manual to  
LL API cross  
reference:
- JSQR JSQ1 LL\_ADC\_INJ\_SetSequencerRanks
  - JSQR JSQ2 LL\_ADC\_INJ\_SetSequencerRanks
  - JSQR JSQ3 LL\_ADC\_INJ\_SetSequencerRanks
  - JSQR JSQ4 LL\_ADC\_INJ\_SetSequencerRanks

### **LL\_ADC\_INJ\_GetSequencerRanks**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_ADC_INJ_GetSequencerRanks(ADC_TypeDef * ADCx, uint32_t Rank)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Get ADC group injected sequence: channel on the selected sequence rank.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Rank:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_ADC_INJ_RANK_1</li> <li>– LL_ADC_INJ_RANK_2</li> <li>– LL_ADC_INJ_RANK_3</li> <li>– LL_ADC_INJ_RANK_4</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (1) On STM32F3, parameter available only on ADC instance: ADC1. <ul style="list-style-type: none"> <li>– LL_ADC_CHANNEL_0</li> <li>– LL_ADC_CHANNEL_1</li> <li>– LL_ADC_CHANNEL_2</li> <li>– LL_ADC_CHANNEL_3</li> <li>– LL_ADC_CHANNEL_4</li> <li>– LL_ADC_CHANNEL_5</li> <li>– LL_ADC_CHANNEL_6</li> <li>– LL_ADC_CHANNEL_7</li> <li>– LL_ADC_CHANNEL_8</li> <li>– LL_ADC_CHANNEL_9</li> <li>– LL_ADC_CHANNEL_10</li> <li>– LL_ADC_CHANNEL_11</li> <li>– LL_ADC_CHANNEL_12</li> <li>– LL_ADC_CHANNEL_13</li> <li>– LL_ADC_CHANNEL_14</li> <li>– LL_ADC_CHANNEL_15</li> <li>– LL_ADC_CHANNEL_16</li> <li>– LL_ADC_CHANNEL_17</li> <li>– LL_ADC_CHANNEL_18</li> <li>– LL_ADC_CHANNEL_VREFINT (5)</li> <li>– LL_ADC_CHANNEL_TEMPSENSOR (1)</li> <li>– LL_ADC_CHANNEL_VBAT (1)</li> <li>– LL_ADC_CHANNEL_VOPAMP1 (1)</li> <li>– LL_ADC_CHANNEL_VOPAMP2 (2)</li> <li>– LL_ADC_CHANNEL_VOPAMP3 (3)</li> </ul> </li> </ul> |

- LL\_ADC\_CHANNEL\_VOPAMP4 (4)
  - (2) On STM32F3, parameter available only on ADC instance: ADC2.
  - (3) On STM32F3, parameter available only on ADC instance: ADC3.
  - (4) On STM32F3, parameter available only on ADC instances: ADC4.
  - (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.
  - (1, 2, 3, 4, 5) For ADC channel read back from ADC register, comparison with internal channel parameter to be done using helper macro `__LL_ADC_CHANNEL_INTERNAL_TO_EXTERNAL()`.
- Notes**
- Depending on devices and packages, some channels may not be available. Refer to device datasheet for channels availability.
  - Usage of the returned channel number: To reinject this channel into another function LL\_ADC\_xxx: the returned channel number is only partly formatted on definition of literals LL\_ADC\_CHANNEL\_x. Therefore, it has to be compared with parts of literals LL\_ADC\_CHANNEL\_x or using helper macro `__LL_ADC_CHANNEL_TO_DECIMAL_NB()`. Then the selected literal LL\_ADC\_CHANNEL\_x can be used as parameter for another function. To get the channel number in decimal format: process the returned value with the helper macro `__LL_ADC_CHANNEL_TO_DECIMAL_NB()`.
- Reference Manual to LL API cross reference:**
- JSQR JSQ1 LL\_ADC\_INJ\_SetSequencerRanks
  - JSQR JSQ2 LL\_ADC\_INJ\_SetSequencerRanks
  - JSQR JSQ3 LL\_ADC\_INJ\_SetSequencerRanks
  - JSQR JSQ4 LL\_ADC\_INJ\_SetSequencerRanks

## LL\_ADC\_INJ\_SetTrigAuto

|                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>        | <code>__STATIC_INLINE void LL_ADC_INJ_SetTrigAuto(ADC_TypeDef * ADCx, uint32_t TrigAuto)</code>                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Function description</b> | Set ADC group injected conversion trigger: independent or from ADC group regular.                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Parameters</b>           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>TrigAuto:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_INJ_TRIG_INDEPENDENT</li> <li>- LL_ADC_INJ_TRIG_FROM_GRP_REGULAR</li> </ul> </li> </ul>                                                                                                                                                                                    |
| <b>Return values</b>        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Notes</b>                | <ul style="list-style-type: none"> <li>• This mode can be used to extend number of data registers updated after one ADC conversion trigger and with data permanently kept (not erased by successive conversions of scan of ADC sequencer ranks), up to 5 data registers: 1 data register on ADC group regular, 4 data registers on ADC group injected.</li> <li>• If ADC group injected trigger source is set to an external trigger, this feature must be set to</li> </ul> |

independent trigger. ADC group injected automatic trigger is compliant only with group injected trigger source set to SW start, without any further action on ADC group injected conversion start or stop: in this case, ADC group injected is controlled only from ADC group regular.

- It is not possible to enable both ADC group injected auto-injected mode and sequencer discontinuous mode.
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.

Reference Manual to  
LL API cross  
reference:

- CFGR JAUTO LL\_ADC\_INJ\_SetTrigAuto

### **LL\_ADC\_INJ\_GetTrigAuto**

|                                                   |                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_INJ_GetTrigAuto(ADC_TypeDef * ADCx)</code>                                                                                                                                                                      |
| Function description                              | Get ADC group injected conversion trigger: independent or from ADC group regular.                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_ADC_INJ_TRIG_INDEPENDENT</li> <li>- LL_ADC_INJ_TRIG_FROM_GRP_REGULAR</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR JAUTO LL_ADC_INJ_SetTrigAuto</li> </ul>                                                                                                                                                                 |

### **LL\_ADC\_INJ\_SetQueueMode**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_INJ_SetQueueMode(ADC_TypeDef * ADCx, uint32_t QueueMode)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Set ADC group injected contexts queue mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>QueueMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_ADC_INJ_QUEUE_2CONTEXTS_LAST_ACTIVE</li> <li>- LL_ADC_INJ_QUEUE_2CONTEXTS_END_EMPTY</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• A context is a setting of group injected sequencer: group injected triggersequencer lengthsequencer ranks If contexts queue is disabled:only 1 sequence can be configured and is active perpetually. If contexts queue is enabled:up to 2 contexts can be queued and are checked in and out as a FIFO stack (first-in, first-out).If a new context is set when queues is full, error is triggered by interruption "Injected Queue Overflow".Two behaviors are possible when all contexts have been processed: the contexts queue can maintain the last context active perpetually or can be empty and injected group triggers are disabled.Triggers can be only external (not internal)</li> </ul> |

SW start)Caution: The sequence must be fully configured in one time (one write of register JSQR makes a check-in of a new context into the queue). Therefore functions to set separately injected trigger and sequencer channels cannot be used, register JSQR must be set using function LL\_ADC\_INJ\_ConfigQueueContext().

- This parameter can be modified only when no conversion is on going on either groups regular or injected.
- A modification of the context mode (bit JQDIS) causes the contexts queue to be flushed and the register JSQR is cleared.
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.
- CFGR JQM LL\_ADC\_INJ\_SetQueueMode

Reference Manual  
to LL API cross  
reference:

### **LL\_ADC\_INJ\_GetQueueMode**

|                                                   |                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_ADC_INJ_GetQueueMode(ADC_TypeDef * ADCx)</code></b>                                                                                                                                                                              |
| Function description                              | Get ADC group injected context queue mode.                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_ADC_INJ_QUEUE_2CONTEXTS_LAST_ACTIVE</li> <li>- LL_ADC_INJ_QUEUE_2CONTEXTS_END_EMPTY</li> </ul> </li> </ul> |
| Reference Manual<br>to LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR JQM LL_ADC_INJ_GetQueueMode</li> </ul>                                                                                                                                                                                 |

### **LL\_ADC\_INJ\_ConfigQueueContext**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_ADC_INJ_ConfigQueueContext(ADC_TypeDef * ADCx, uint32_t TriggerSource, uint32_t ExternalTriggerEdge, uint32_t SequencerNbRanks, uint32_t Rank1_Channel, uint32_t Rank2_Channel, uint32_t Rank3_Channel, uint32_t Rank4_Channel)</code></b>                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description | Set one context on ADC group injected that will be checked in contexts queue.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>TriggerSource:</b> This parameter can be one of the following values: (1) On STM32F3, parameter not available on all devices: among others, on STM32F303xE, STM32F398xx.           <ul style="list-style-type: none"> <li>- LL_ADC_INJ_TRIG_SOFTWARE</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM1_TRGO</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM1_TRGO2</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM1_CH3_ADC34 (1)(2) (8)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM1_CH4</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM2_TRGO (3)(4)(5)</li> <li>- LL_ADC_INJ_TRIG_EXT_TIM2_TRGO_ADC12 (1)(2) (7)</li> </ul> </li> </ul> |

- LL\_ADC\_INJ\_TRIG\_EXT\_TIM2\_TRGO\_ADC34 (1)(2)  
(8)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM2\_CH1 (3)(4)(5)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM2\_CH1\_ADC12 (1)(2) (7)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM3\_TRGO (1)(2)(3)(4)(5)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM3\_CH1 (3)(4)(5)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM3\_CH1\_ADC12 (1)(2) (7)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM3\_CH3 (3)(4)(5)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM3\_CH3\_ADC12 (1)(2) (7)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM3\_CH4 (3)(4)(5)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM3\_CH4\_ADC12  
(1)(2)(3)(4)(5) (7)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM4\_TRGO (3) (5)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM4\_TRGO\_ADC12 (1)(2)  
(7)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM4\_TRGO\_ADC34 (1)(2)  
(8)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM4\_CH3\_ADC34 (1)(2) (8)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM4\_CH4\_ADC34 (1)(2) (8)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM6\_TRGO (3)(4)(5)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM6\_TRGO\_ADC12 (1)(2)  
(7)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM7\_TRGO\_ADC34 (1)(2)  
(8)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM8\_TRGO (1)(2)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM8\_TRGO2 (1)(2)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM8\_CH2\_ADC34 (1)(2) (8)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM8\_CH4\_ADC12 (1)(2) (7)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM8\_CH4\_ADC34 (1)(2) (8)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM15\_TRGO
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRGO\_ADC12 (1) (7)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRGO2\_ADC12 (1)  
(7)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_CH4\_ADC12 (1) (7)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRG\_ADC34 (1) (8)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_TRG2\_ADC34 (1) (8)
  - LL\_ADC\_INJ\_TRIG\_EXT\_TIM20\_CH2\_ADC34 (1) (8)
  - LL\_ADC\_INJ\_TRIG\_EXT\_HRTIM\_TRG2 (4)
  - LL\_ADC\_INJ\_TRIG\_EXT\_HRTIM\_TRG4 (4)
  - LL\_ADC\_INJ\_TRIG\_EXT\_EXTI\_LINE15 (3)(4)(5)(6)
  - LL\_ADC\_INJ\_TRIG\_EXT\_EXTI\_LINE15\_ADC12 (1)(2)  
(7)
- (2) On STM32F3, parameter not available on all devices:  
among others, on STM32F303xC, STM32F358xx.
  - (3) On STM32F3, parameter not available on all devices:  
among others, on STM32F303x8, STM32F328xx.
  - (4) On STM32F3, parameter not available on all devices:  
among others, on STM32F334x8.
  - (5) On STM32F3, parameter not available on all devices:  
among others, on STM32F302xC, STM32F302xE.
  - (6) On STM32F3, parameter not available on all devices:  
among others, on STM32F301x8, STM32F302x8,  
STM32F318xx.

- (7) On STM32F3, parameter not available on all ADC instances: ADC1, ADC2 (for ADC instances ADCx available on the selected device).
- (8) On STM32F3, parameter not available on all ADC instances: ADC3, ADC4 (for ADC instances ADCx available on the selected device).
- **ExternalTriggerEdge:** This parameter can be one of the following values: Note: This parameter is discarded in case of SW start: parameter "TriggerSource" set to "LL\_ADC\_INJ\_TRIG\_SOFTWARE".
  - LL\_ADC\_INJ\_TRIG\_EXT\_RISING
  - LL\_ADC\_INJ\_TRIG\_EXT\_FALLING
  - LL\_ADC\_INJ\_TRIG\_EXT\_RISINGFALLING
- **SequencerNbRanks:** This parameter can be one of the following values:
  - LL\_ADC\_INJ\_SEQ\_SCAN\_DISABLE
  - LL\_ADC\_INJ\_SEQ\_SCAN\_ENABLE\_2RANKS
  - LL\_ADC\_INJ\_SEQ\_SCAN\_ENABLE\_3RANKS
  - LL\_ADC\_INJ\_SEQ\_SCAN\_ENABLE\_4RANKS
- **Rank1\_Channel:** This parameter can be one of the following values: (1) On STM32F3, parameter available only on ADC instance: ADC1.
  - LL\_ADC\_CHANNEL\_0
  - LL\_ADC\_CHANNEL\_1
  - LL\_ADC\_CHANNEL\_2
  - LL\_ADC\_CHANNEL\_3
  - LL\_ADC\_CHANNEL\_4
  - LL\_ADC\_CHANNEL\_5
  - LL\_ADC\_CHANNEL\_6
  - LL\_ADC\_CHANNEL\_7
  - LL\_ADC\_CHANNEL\_8
  - LL\_ADC\_CHANNEL\_9
  - LL\_ADC\_CHANNEL\_10
  - LL\_ADC\_CHANNEL\_11
  - LL\_ADC\_CHANNEL\_12
  - LL\_ADC\_CHANNEL\_13
  - LL\_ADC\_CHANNEL\_14
  - LL\_ADC\_CHANNEL\_15
  - LL\_ADC\_CHANNEL\_16
  - LL\_ADC\_CHANNEL\_17
  - LL\_ADC\_CHANNEL\_18
  - LL\_ADC\_CHANNEL\_VREFINT (5)
  - LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
  - LL\_ADC\_CHANNEL\_VBAT (1)
  - LL\_ADC\_CHANNEL\_VOPAMP1 (1)
  - LL\_ADC\_CHANNEL\_VOPAMP2 (2)
  - LL\_ADC\_CHANNEL\_VOPAMP3 (3)
  - LL\_ADC\_CHANNEL\_VOPAMP4 (4)
- (2) On STM32F3, parameter available only on ADC instance: ADC2.
- (3) On STM32F3, parameter available only on ADC instance: ADC3.
- (4) On STM32F3, parameter available only on ADC

- instances: ADC4.
- (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.
  - **Rank2\_Channel:** This parameter can be one of the following values: (1) On STM32F3, parameter available only on ADC instance: ADC1.
    - LL\_ADC\_CHANNEL\_0
    - LL\_ADC\_CHANNEL\_1
    - LL\_ADC\_CHANNEL\_2
    - LL\_ADC\_CHANNEL\_3
    - LL\_ADC\_CHANNEL\_4
    - LL\_ADC\_CHANNEL\_5
    - LL\_ADC\_CHANNEL\_6
    - LL\_ADC\_CHANNEL\_7
    - LL\_ADC\_CHANNEL\_8
    - LL\_ADC\_CHANNEL\_9
    - LL\_ADC\_CHANNEL\_10
    - LL\_ADC\_CHANNEL\_11
    - LL\_ADC\_CHANNEL\_12
    - LL\_ADC\_CHANNEL\_13
    - LL\_ADC\_CHANNEL\_14
    - LL\_ADC\_CHANNEL\_15
    - LL\_ADC\_CHANNEL\_16
    - LL\_ADC\_CHANNEL\_17
    - LL\_ADC\_CHANNEL\_18
    - LL\_ADC\_CHANNEL\_VREFINT (5)
    - LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
    - LL\_ADC\_CHANNEL\_VBAT (1)
    - LL\_ADC\_CHANNEL\_VOPAMP1 (1)
    - LL\_ADC\_CHANNEL\_VOPAMP2 (2)
    - LL\_ADC\_CHANNEL\_VOPAMP3 (3)
    - LL\_ADC\_CHANNEL\_VOPAMP4 (4)
  - (2) On STM32F3, parameter available only on ADC instance: ADC2.
  - (3) On STM32F3, parameter available only on ADC instance: ADC3.
  - (4) On STM32F3, parameter available only on ADC instances: ADC4.
  - (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.
  - **Rank3\_Channel:** This parameter can be one of the following values: (1) On STM32F3, parameter available only on ADC instance: ADC1.
    - LL\_ADC\_CHANNEL\_0
    - LL\_ADC\_CHANNEL\_1
    - LL\_ADC\_CHANNEL\_2
    - LL\_ADC\_CHANNEL\_3
    - LL\_ADC\_CHANNEL\_4
    - LL\_ADC\_CHANNEL\_5
    - LL\_ADC\_CHANNEL\_6
    - LL\_ADC\_CHANNEL\_7

- LL\_ADC\_CHANNEL\_8
- LL\_ADC\_CHANNEL\_9
- LL\_ADC\_CHANNEL\_10
- LL\_ADC\_CHANNEL\_11
- LL\_ADC\_CHANNEL\_12
- LL\_ADC\_CHANNEL\_13
- LL\_ADC\_CHANNEL\_14
- LL\_ADC\_CHANNEL\_15
- LL\_ADC\_CHANNEL\_16
- LL\_ADC\_CHANNEL\_17
- LL\_ADC\_CHANNEL\_18
- LL\_ADC\_CHANNEL\_VREFINT (5)
- LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
- LL\_ADC\_CHANNEL\_VBAT (1)
- LL\_ADC\_CHANNEL\_VOPAMP1 (1)
- LL\_ADC\_CHANNEL\_VOPAMP2 (2)
- LL\_ADC\_CHANNEL\_VOPAMP3 (3)
- LL\_ADC\_CHANNEL\_VOPAMP4 (4)
- (2) On STM32F3, parameter available only on ADC instance: ADC2.
- (3) On STM32F3, parameter available only on ADC instance: ADC3.
- (4) On STM32F3, parameter available only on ADC instances: ADC4.
- (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.
- **Rank4\_Channel:** This parameter can be one of the following values: (1) On STM32F3, parameter available only on ADC instance: ADC1.
  - LL\_ADC\_CHANNEL\_0
  - LL\_ADC\_CHANNEL\_1
  - LL\_ADC\_CHANNEL\_2
  - LL\_ADC\_CHANNEL\_3
  - LL\_ADC\_CHANNEL\_4
  - LL\_ADC\_CHANNEL\_5
  - LL\_ADC\_CHANNEL\_6
  - LL\_ADC\_CHANNEL\_7
  - LL\_ADC\_CHANNEL\_8
  - LL\_ADC\_CHANNEL\_9
  - LL\_ADC\_CHANNEL\_10
  - LL\_ADC\_CHANNEL\_11
  - LL\_ADC\_CHANNEL\_12
  - LL\_ADC\_CHANNEL\_13
  - LL\_ADC\_CHANNEL\_14
  - LL\_ADC\_CHANNEL\_15
  - LL\_ADC\_CHANNEL\_16
  - LL\_ADC\_CHANNEL\_17
  - LL\_ADC\_CHANNEL\_18
  - LL\_ADC\_CHANNEL\_VREFINT (5)
  - LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
  - LL\_ADC\_CHANNEL\_VBAT (1)
  - LL\_ADC\_CHANNEL\_VOPAMP1 (1)

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                             | <ul style="list-style-type: none"> <li>- LL_ADC_CHANNEL_VOPAMP2 (2)</li> <li>- LL_ADC_CHANNEL_VOPAMP3 (3)</li> <li>- LL_ADC_CHANNEL_VOPAMP4 (4)</li> </ul> <ul style="list-style-type: none"> <li>• (2) On STM32F3, parameter available only on ADC instance: ADC2.</li> <li>• (3) On STM32F3, parameter available only on ADC instance: ADC3.</li> <li>• (4) On STM32F3, parameter available only on ADC instances: ADC4.</li> <li>• (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Return values                               | • <b>None</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                                       | <ul style="list-style-type: none"> <li>• A context is a setting of group injected sequencer: group injected triggersequencer lengthsequencer ranks This function is intended to be used when contexts queue is enabled, because the sequence must be fully configured in one time (functions to set separately injected trigger and sequencer channels cannot be used): Refer to function LL_ADC_INJ_SetQueueMode().</li> <li>• In the contexts queue, only the active context can be read. The parameters of this function can be read using functions: LL_ADC_INJ_GetTriggerSource() LL_ADC_INJ_GetTriggerEdge() LL_ADC_INJ_GetSequencerRanks()</li> <li>• On this STM32 serie, to measure internal channels (VrefInt, TempSensor, ...), measurement paths to internal channels must be enabled separately. This can be done using function LL_ADC_SetCommonPathInternalCh().</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must not be disabled. Can be enabled with or without conversion on going on either groups regular or injected.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• JSQR JEXTSEL LL_ADC_INJ_ConfigQueueContext</li> <li>• JSQR JEXTEN LL_ADC_INJ_ConfigQueueContext</li> <li>• JSQR JL LL_ADC_INJ_ConfigQueueContext</li> <li>• JSQR JSQ1 LL_ADC_INJ_ConfigQueueContext</li> <li>• JSQR JSQ2 LL_ADC_INJ_ConfigQueueContext</li> <li>• JSQR JSQ3 LL_ADC_INJ_ConfigQueueContext</li> <li>• JSQR JSQ4 LL_ADC_INJ_ConfigQueueContext</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

## LL\_ADC\_SetChannelSamplingTime

|                      |                                                                                                                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>STATIC_INLINE void LL_ADC_SetChannelSamplingTime (ADC_TypeDef * ADCx, uint32_t Channel, uint32_t SamplingTime)</b>                                                                                                                                                    |
| Function description | Set sampling time of the selected ADC channel Unit: ADC clock cycles.                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>(1) On STM32F3, parameter available only on ADC instance:</li> </ul> </li> </ul> |

## ADC1.

- LL\_ADC\_CHANNEL\_0
- LL\_ADC\_CHANNEL\_1
- LL\_ADC\_CHANNEL\_2
- LL\_ADC\_CHANNEL\_3
- LL\_ADC\_CHANNEL\_4
- LL\_ADC\_CHANNEL\_5
- LL\_ADC\_CHANNEL\_6
- LL\_ADC\_CHANNEL\_7
- LL\_ADC\_CHANNEL\_8
- LL\_ADC\_CHANNEL\_9
- LL\_ADC\_CHANNEL\_10
- LL\_ADC\_CHANNEL\_11
- LL\_ADC\_CHANNEL\_12
- LL\_ADC\_CHANNEL\_13
- LL\_ADC\_CHANNEL\_14
- LL\_ADC\_CHANNEL\_15
- LL\_ADC\_CHANNEL\_16
- LL\_ADC\_CHANNEL\_17
- LL\_ADC\_CHANNEL\_18
- LL\_ADC\_CHANNEL\_VREFINT (5)
- LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
- LL\_ADC\_CHANNEL\_VBAT (1)
- LL\_ADC\_CHANNEL\_VOPAMP1 (1)
- LL\_ADC\_CHANNEL\_VOPAMP2 (2)
- LL\_ADC\_CHANNEL\_VOPAMP3 (3)
- LL\_ADC\_CHANNEL\_VOPAMP4 (4)

- (2) On STM32F3, parameter available only on ADC instance: ADC2.
- (3) On STM32F3, parameter available only on ADC instance: ADC3.
- (4) On STM32F3, parameter available only on ADC instances: ADC4.
- (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.
- **SamplingTime:** This parameter can be one of the following values:
  - LL\_ADC\_SAMPLINGTIME\_1CYCLE\_5
  - LL\_ADC\_SAMPLINGTIME\_2CYCLES\_5
  - LL\_ADC\_SAMPLINGTIME\_4CYCLES\_5
  - LL\_ADC\_SAMPLINGTIME\_7CYCLES\_5
  - LL\_ADC\_SAMPLINGTIME\_19CYCLES\_5
  - LL\_ADC\_SAMPLINGTIME\_61CYCLES\_5
  - LL\_ADC\_SAMPLINGTIME\_181CYCLES\_5
  - LL\_ADC\_SAMPLINGTIME\_601CYCLES\_5

## Return values

- **None**

## Notes

- On this device, sampling time is on channel scope: independently of channel mapped on ADC group regular or injected.
- In case of internal channel (VrefInt, TempSensor, ...) to be converted: sampling time constraints must be respected

(sampling time can be adjusted in function of ADC clock frequency and sampling time setting). Refer to device datasheet for timings values (parameters TS\_vrefint, TS\_temp, ...).

- Conversion time is the addition of sampling time and processing time. On this STM32 serie, ADC processing time is: 12.5 ADC clock cycles at ADC resolution 12 bits10.5 ADC clock cycles at ADC resolution 10 bits8.5 ADC clock cycles at ADC resolution 8 bits6.5 ADC clock cycles at ADC resolution 6 bits
- In case of ADC conversion of internal channel (VrefInt, temperature sensor, ...), a sampling time minimum value is required. Refer to device datasheet.
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.

Reference Manual to  
LL API cross  
reference:

- SMPR1 SMP0 LL\_ADC\_SetChannelSamplingTime
- SMPR1 SMP1 LL\_ADC\_SetChannelSamplingTime
- SMPR1 SMP2 LL\_ADC\_SetChannelSamplingTime
- SMPR1 SMP3 LL\_ADC\_SetChannelSamplingTime
- SMPR1 SMP4 LL\_ADC\_SetChannelSamplingTime
- SMPR1 SMP5 LL\_ADC\_SetChannelSamplingTime
- SMPR1 SMP6 LL\_ADC\_SetChannelSamplingTime
- SMPR1 SMP7 LL\_ADC\_SetChannelSamplingTime
- SMPR1 SMP8 LL\_ADC\_SetChannelSamplingTime
- SMPR1 SMP9 LL\_ADC\_SetChannelSamplingTime
- SMPR2 SMP10 LL\_ADC\_SetChannelSamplingTime
- SMPR2 SMP11 LL\_ADC\_SetChannelSamplingTime
- SMPR2 SMP12 LL\_ADC\_SetChannelSamplingTime
- SMPR2 SMP13 LL\_ADC\_SetChannelSamplingTime
- SMPR2 SMP14 LL\_ADC\_SetChannelSamplingTime
- SMPR2 SMP15 LL\_ADC\_SetChannelSamplingTime
- SMPR2 SMP16 LL\_ADC\_SetChannelSamplingTime
- SMPR2 SMP17 LL\_ADC\_SetChannelSamplingTime
- SMPR2 SMP18 LL\_ADC\_SetChannelSamplingTime

### **LL\_ADC\_GetChannelSamplingTime**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_GetChannelSamplingTime (ADC_TypeDef * ADCx,<br/>uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                    |
| Function description | Get sampling time of the selected ADC channel Unit: ADC clock cycles.                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values:<br/>(1) On STM32F3, parameter available only on ADC instance: ADC1. <ul style="list-style-type: none"> <li>- LL_ADC_CHANNEL_0</li> <li>- LL_ADC_CHANNEL_1</li> <li>- LL_ADC_CHANNEL_2</li> <li>- LL_ADC_CHANNEL_3</li> <li>- LL_ADC_CHANNEL_4</li> </ul> </li> </ul> |

- LL\_ADC\_CHANNEL\_5
  - LL\_ADC\_CHANNEL\_6
  - LL\_ADC\_CHANNEL\_7
  - LL\_ADC\_CHANNEL\_8
  - LL\_ADC\_CHANNEL\_9
  - LL\_ADC\_CHANNEL\_10
  - LL\_ADC\_CHANNEL\_11
  - LL\_ADC\_CHANNEL\_12
  - LL\_ADC\_CHANNEL\_13
  - LL\_ADC\_CHANNEL\_14
  - LL\_ADC\_CHANNEL\_15
  - LL\_ADC\_CHANNEL\_16
  - LL\_ADC\_CHANNEL\_17
  - LL\_ADC\_CHANNEL\_18
  - LL\_ADC\_CHANNEL\_VREFINT (5)
  - LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
  - LL\_ADC\_CHANNEL\_VBAT (1)
  - LL\_ADC\_CHANNEL\_VOPAMP1 (1)
  - LL\_ADC\_CHANNEL\_VOPAMP2 (2)
  - LL\_ADC\_CHANNEL\_VOPAMP3 (3)
  - LL\_ADC\_CHANNEL\_VOPAMP4 (4)
  - (2) On STM32F3, parameter available only on ADC instance: ADC2.
  - (3) On STM32F3, parameter available only on ADC instance: ADC3.
  - (4) On STM32F3, parameter available only on ADC instances: ADC4.
  - (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time.
- Return values**
- **Returned:** value can be one of the following values:
    - LL\_ADC\_SAMPLINGTIME\_1CYCLE\_5
    - LL\_ADC\_SAMPLINGTIME\_2CYCLES\_5
    - LL\_ADC\_SAMPLINGTIME\_4CYCLES\_5
    - LL\_ADC\_SAMPLINGTIME\_7CYCLES\_5
    - LL\_ADC\_SAMPLINGTIME\_19CYCLES\_5
    - LL\_ADC\_SAMPLINGTIME\_61CYCLES\_5
    - LL\_ADC\_SAMPLINGTIME\_181CYCLES\_5
    - LL\_ADC\_SAMPLINGTIME\_601CYCLES\_5
- Notes**
- On this device, sampling time is on channel scope: independently of channel mapped on ADC group regular or injected.
  - Conversion time is the addition of sampling time and processing time. On this STM32 serie, ADC processing time is: 12.5 ADC clock cycles at ADC resolution 12 bits10.5 ADC clock cycles at ADC resolution 10 bits8.5 ADC clock cycles at ADC resolution 8 bits6.5 ADC clock cycles at ADC resolution 6 bits
- Reference Manual to  
LL API cross  
reference:**
- SMPR1 SMP0 LL\_ADC\_GetChannelSamplingTime
  - SMPR1 SMP1 LL\_ADC\_GetChannelSamplingTime
  - SMPR1 SMP2 LL\_ADC\_GetChannelSamplingTime

- SMPR1 SMP3 LL\_ADC\_GetChannelSamplingTime
- SMPR1 SMP4 LL\_ADC\_GetChannelSamplingTime
- SMPR1 SMP5 LL\_ADC\_GetChannelSamplingTime
- SMPR1 SMP6 LL\_ADC\_GetChannelSamplingTime
- SMPR1 SMP7 LL\_ADC\_GetChannelSamplingTime
- SMPR1 SMP8 LL\_ADC\_GetChannelSamplingTime
- SMPR1 SMP9 LL\_ADC\_GetChannelSamplingTime
- SMPR2 SMP10 LL\_ADC\_GetChannelSamplingTime
- SMPR2 SMP11 LL\_ADC\_GetChannelSamplingTime
- SMPR2 SMP12 LL\_ADC\_GetChannelSamplingTime
- SMPR2 SMP13 LL\_ADC\_GetChannelSamplingTime
- SMPR2 SMP14 LL\_ADC\_GetChannelSamplingTime
- SMPR2 SMP15 LL\_ADC\_GetChannelSamplingTime
- SMPR2 SMP16 LL\_ADC\_GetChannelSamplingTime
- SMPR2 SMP17 LL\_ADC\_GetChannelSamplingTime
- SMPR2 SMP18 LL\_ADC\_GetChannelSamplingTime

### **LL\_ADC\_SetChannelSingleDiff**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_SetChannelSingleDiff(ADC_TypeDef * ADCx, uint32_t Channel, uint32_t SingleDiff)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description | Set mode single-ended or differential input of the selected ADC channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values:<br/>(1) On STM32F3, parameter available only on ADC instance: ADC1. <ul style="list-style-type: none"> <li>– LL_ADC_CHANNEL_1</li> <li>– LL_ADC_CHANNEL_2</li> <li>– LL_ADC_CHANNEL_3</li> <li>– LL_ADC_CHANNEL_4</li> <li>– LL_ADC_CHANNEL_5</li> <li>– LL_ADC_CHANNEL_6</li> <li>– LL_ADC_CHANNEL_7</li> <li>– LL_ADC_CHANNEL_8</li> <li>– LL_ADC_CHANNEL_9</li> <li>– LL_ADC_CHANNEL_10</li> <li>– LL_ADC_CHANNEL_11</li> <li>– LL_ADC_CHANNEL_12</li> <li>– LL_ADC_CHANNEL_13</li> <li>– LL_ADC_CHANNEL_14</li> <li>– LL_ADC_CHANNEL_15</li> <li>– LL_ADC_CHANNEL_16 (1)</li> </ul> </li> <li>• <b>SingleDiff:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_ADC_SINGLE_ENDED</li> <li>– LL_ADC_DIFFERENTIAL_ENDED</li> </ul> </li> <li>• <b>None</b></li> </ul> |
| Return values        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• Channel ending is on channel scope: independently of channel mapped on ADC group regular or injected. In differential mode: Differential measurement is carried out</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

between the selected channel 'i' (positive input) and channel 'i+1' (negative input). Only channel 'i' has to be configured, channel 'i+1' is configured automatically.

- Refer to Reference Manual to ensure the selected channel is available in differential mode. For example, internal channels (VrefInt, TempSensor, ...) are not available in differential mode.
- When configuring a channel 'i' in differential mode, the channel 'i+1' is not usable separately.
- On STM32F3, channels 16, 17, 18 of ADC1, channels 17, 18 of ADC2, ADC3, ADC4 (if available) are internally fixed to single-ended inputs configuration.
- For ADC channels configured in differential mode, both inputs should be biased at  $(Vref+)/2 \pm 200\text{mV}$ . ( $Vref+$  is the analog voltage reference)
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be ADC disabled.
- One or several values can be selected. Example: (LL\_ADC\_CHANNEL\_4 | LL\_ADC\_CHANNEL\_12 | ...)
- DIFSEL DIFSEL LL\_ADC\_GetChannelSamplingTime

Reference Manual to  
LL API cross  
reference:

### LL\_ADC\_GetChannelSingleDiff

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_GetChannelSingleDiff(ADC_TypeDef * ADCx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Get mode single-ended or differential input of the selected ADC channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Channel:</b> This parameter can be a combination of the following values: (1) On STM32F3, parameter available only on ADC instance: ADC1. <ul style="list-style-type: none"> <li>– LL_ADC_CHANNEL_0</li> <li>– LL_ADC_CHANNEL_1</li> <li>– LL_ADC_CHANNEL_2</li> <li>– LL_ADC_CHANNEL_3</li> <li>– LL_ADC_CHANNEL_4</li> <li>– LL_ADC_CHANNEL_5</li> <li>– LL_ADC_CHANNEL_6</li> <li>– LL_ADC_CHANNEL_7</li> <li>– LL_ADC_CHANNEL_8</li> <li>– LL_ADC_CHANNEL_9</li> <li>– LL_ADC_CHANNEL_10</li> <li>– LL_ADC_CHANNEL_11</li> <li>– LL_ADC_CHANNEL_12</li> <li>– LL_ADC_CHANNEL_13</li> <li>– LL_ADC_CHANNEL_14</li> <li>– LL_ADC_CHANNEL_15</li> <li>– LL_ADC_CHANNEL_16 (1)</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>0:</b> channel in single-ended mode, else: channel in differential</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | mode                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Notes                                             | <ul style="list-style-type: none"> <li>When configuring a channel 'i' in differential mode, the channel 'i+1' is not usable separately. Therefore, to ensure a channel is configured in single-ended mode, the configuration of channel itself and the channel 'i-1' must be read back (to ensure that the selected channel channel has not been configured in differential mode by the previous channel).</li> <li>Refer to Reference Manual to ensure the selected channel is available in differential mode. For example, internal channels (VrefInt, TempSensor, ...) are not available in differential mode.</li> <li>When configuring a channel 'i' in differential mode, the channel 'i+1' is not usable separately.</li> <li>On STM32F3, channels 16, 17, 18 of ADC1, channels 17, 18 of ADC2, ADC3, ADC4 (if available) are internally fixed to single-ended inputs configuration.</li> <li>One or several values can be selected. In this case, the value returned is null if all channels are in single ended-mode.<br/>Example: (LL_ADC_CHANNEL_4   LL_ADC_CHANNEL_12   ...)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>DIFSEL DIFSEL LL_ADC_GetChannelSamplingTime</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

## LL\_ADC\_SetAnalogWDMonitChannels

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_SetAnalogWDMonitChannels(ADC_TypeDef * ADCx, uint32_t AWDy, uint32_t AWDChannelGroup)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description | Set ADC analog watchdog monitored channels: a single channel, multiple channels or all channels, on ADC groups regular and-or injected.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> <li><b>AWDy:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_AWD1</li> <li>- LL_ADC_AWD2</li> <li>- LL_ADC_AWD3</li> </ul> </li> <li><b>AWDChannelGroup:</b> This parameter can be one of the following values: (0) On STM32F3, parameter available only on analog watchdog number: AWD1. <ul style="list-style-type: none"> <li>- LL_ADC_AWD_DISABLE</li> <li>- LL_ADC_AWD_ALL_CHANNELS_REG (0)</li> <li>- LL_ADC_AWD_ALL_CHANNELS_INJ (0)</li> <li>- LL_ADC_AWD_ALL_CHANNELS_REG_INJ</li> <li>- LL_ADC_AWD_CHANNEL_0_REG (0)</li> <li>- LL_ADC_AWD_CHANNEL_0_INJ (0)</li> <li>- LL_ADC_AWD_CHANNEL_0_REG_INJ</li> <li>- LL_ADC_AWD_CHANNEL_1_REG (0)</li> <li>- LL_ADC_AWD_CHANNEL_1_INJ (0)</li> <li>- LL_ADC_AWD_CHANNEL_1_REG_INJ</li> <li>- LL_ADC_AWD_CHANNEL_2_REG (0)</li> <li>- LL_ADC_AWD_CHANNEL_2_INJ (0)</li> </ul> </li> </ul> |

- LL\_ADC\_AWD\_CHANNEL\_2\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_3\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_3\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_3\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_4\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_4\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_4\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_5\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_5\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_5\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_6\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_6\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_6\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_7\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_7\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_7\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_8\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_8\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_8\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_9\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_9\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_9\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_10\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_10\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_10\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_11\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_11\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_11\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_12\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_12\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_12\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_13\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_13\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_13\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_14\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_14\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_14\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_15\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_15\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_15\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_16\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_16\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_16\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_17\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_17\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_17\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_18\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_18\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_18\_REG\_INJ
- LL\_ADC\_AWD\_CH\_VREFINT\_REG (0)(5)
- LL\_ADC\_AWD\_CH\_VREFINT\_INJ (0)(5)
- LL\_ADC\_AWD\_CH\_VREFINT\_REG\_INJ (5)

- LL\_ADC\_AWD\_CH\_TEMPSENSOR\_REG (0)(1)
- LL\_ADC\_AWD\_CH\_TEMPSENSOR\_INJ (0)(1)
- LL\_ADC\_AWD\_CH\_TEMPSENSOR\_REG\_INJ (1)
- LL\_ADC\_AWD\_CH\_VBAT\_REG (0)(1)
- LL\_ADC\_AWD\_CH\_VBAT\_INJ (0)(1)
- LL\_ADC\_AWD\_CH\_VBAT\_REG\_INJ (1)
- LL\_ADC\_AWD\_CH\_VOPAMP1\_REG (0)(1)
- LL\_ADC\_AWD\_CH\_VOPAMP1\_INJ (0)(1)
- LL\_ADC\_AWD\_CH\_VOPAMP1\_REG\_INJ (1)
- LL\_ADC\_AWD\_CH\_VOPAMP2\_REG (0)(2)
- LL\_ADC\_AWD\_CH\_VOPAMP2\_INJ (0)(2)
- LL\_ADC\_AWD\_CH\_VOPAMP2\_REG\_INJ (2)
- LL\_ADC\_AWD\_CH\_VOPAMP3\_REG (0)(3)
- LL\_ADC\_AWD\_CH\_VOPAMP3\_INJ (0)(3)
- LL\_ADC\_AWD\_CH\_VOPAMP3\_REG\_INJ (3)
- LL\_ADC\_AWD\_CH\_VOPAMP4\_REG (0)(4)
- LL\_ADC\_AWD\_CH\_VOPAMP4\_INJ (0)(4)
- LL\_ADC\_AWD\_CH\_VOPAMP4\_REG\_INJ (4)
- (1) On STM32F3, parameter available only on ADC instance: ADC1.
- (2) On STM32F3, parameter available only on ADC instance: ADC2.
- (3) On STM32F3, parameter available only on ADC instance: ADC3.
- (4) On STM32F3, parameter available only on ADC instances: ADC4.
- (5) On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to Vrefint at the same time.

**Return values**

- **None**

**Notes**

- Once monitored channels are selected, analog watchdog is enabled.
- In case of need to define a single channel to monitor with analog watchdog from sequencer channel definition, use helper macro `_LL_ADC_ANALOGWD_CHANNEL_GROUP()`.
- On this STM32 serie, there are 2 kinds of analog watchdog instance: AWD standard (instance AWD1): channels monitored: can monitor 1 channel or all channels.groups monitored: ADC groups regular and-or injected.resolution: resolution is not limited (corresponds to ADC resolution configured). AWD flexible (instances AWD2, AWD3): channels monitored: flexible on channels monitored, selection is channel wise, from from 1 to all channels. Specificity of this analog watchdog: Multiple channels can be selected. For example: (`LL_ADC_AWD_CHANNEL4_REG_INJ | LL_ADC_AWD_CHANNEL5_REG_INJ | ...`)groups monitored: not selection possible (monitoring on both groups regular and injected). Channels selected are monitored on groups regular and injected: `LL_ADC_AWD_CHANNELxx_REG_INJ` (do not use parameters `LL_ADC_AWD_CHANNELxx_REG` and `LL_ADC_AWD_CHANNELxx_INJ`)resolution: resolution is limited to 8 bits: if ADC resolution is 12 bits the 4 LSB are

Reference Manual to  
LL API cross  
reference:

- ignored, if ADC resolution is 10 bits the 2 LSB are ignored.
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.
- CFGR AWD1CH LL\_ADC\_SetAnalogWDMonitChannels
- CFGR AWD1SGL LL\_ADC\_SetAnalogWDMonitChannels
- CFGR AWD1EN LL\_ADC\_SetAnalogWDMonitChannels
- CFGR JAWD1EN LL\_ADC\_SetAnalogWDMonitChannels
- AWD2CR AWD2CH LL\_ADC\_SetAnalogWDMonitChannels
- AWD3CR AWD3CH LL\_ADC\_SetAnalogWDMonitChannels

### **LL\_ADC\_GetAnalogWDMonitChannels**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_GetAnalogWDMonitChannels (ADC_TypeDef * ADCx,<br/>uint32_t AWDy)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Get ADC analog watchdog monitored channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> <li><b>AWDy:</b> This parameter can be one of the following values:<br/>(1) On this AWD number, monitored channel can be retrieved if only 1 channel is programmed (or none or all channels). This function cannot retrieve monitored channel if multiple channels are programmed simultaneously by bitfield. <ul style="list-style-type: none"> <li>- LL_ADC_AWD1</li> <li>- LL_ADC_AWD2 (1)</li> <li>- LL_ADC_AWD3 (1)</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Return values        | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values: (0) On STM32F3, parameter available only on analog watchdog number: AWD1. <ul style="list-style-type: none"> <li>- LL_ADC_AWD_DISABLE</li> <li>- LL_ADC_AWD_ALL_CHANNELS_REG (0)</li> <li>- LL_ADC_AWD_ALL_CHANNELS_INJ (0)</li> <li>- LL_ADC_AWD_ALL_CHANNELS_REG_INJ</li> <li>- LL_ADC_AWD_CHANNEL_0_REG (0)</li> <li>- LL_ADC_AWD_CHANNEL_0_INJ (0)</li> <li>- LL_ADC_AWD_CHANNEL_0_REG_INJ</li> <li>- LL_ADC_AWD_CHANNEL_1_REG (0)</li> <li>- LL_ADC_AWD_CHANNEL_1_INJ (0)</li> <li>- LL_ADC_AWD_CHANNEL_1_REG_INJ</li> <li>- LL_ADC_AWD_CHANNEL_2_REG (0)</li> <li>- LL_ADC_AWD_CHANNEL_2_INJ (0)</li> <li>- LL_ADC_AWD_CHANNEL_2_REG_INJ</li> <li>- LL_ADC_AWD_CHANNEL_3_REG (0)</li> <li>- LL_ADC_AWD_CHANNEL_3_INJ (0)</li> <li>- LL_ADC_AWD_CHANNEL_3_REG_INJ</li> <li>- LL_ADC_AWD_CHANNEL_4_REG (0)</li> <li>- LL_ADC_AWD_CHANNEL_4_INJ (0)</li> <li>- LL_ADC_AWD_CHANNEL_4_REG_INJ</li> <li>- LL_ADC_AWD_CHANNEL_5_REG (0)</li> <li>- LL_ADC_AWD_CHANNEL_5_INJ (0)</li> <li>- LL_ADC_AWD_CHANNEL_5_REG_INJ</li> </ul> </li> </ul> |

- LL\_ADC\_AWD\_CHANNEL\_6\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_6\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_6\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_7\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_7\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_7\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_8\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_8\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_8\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_9\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_9\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_9\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_10\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_10\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_10\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_11\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_11\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_11\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_12\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_12\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_12\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_13\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_13\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_13\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_14\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_14\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_14\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_15\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_15\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_15\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_16\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_16\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_16\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_17\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_17\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_17\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_18\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_18\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_18\_REG\_INJ

#### Notes

- Usage of the returned channel number: To reinject this channel into another function LL\_ADC\_xxx: the returned channel number is only partly formatted on definition of literals LL\_ADC\_CHANNEL\_x. Therefore, it has to be compared with parts of literals LL\_ADC\_CHANNEL\_x or using helper macro \_\_LL\_ADC\_CHANNEL\_TO\_DECIMAL\_NB(). Then the selected literal LL\_ADC\_CHANNEL\_x can be used as parameter for another function. To get the channel number in decimal format: process the returned value with the helper macro \_\_LL\_ADC\_CHANNEL\_TO\_DECIMAL\_NB(). Applicable only when the analog watchdog is set to monitor one channel.
- On this STM32 serie, there are 2 kinds of analog watchdog

instance: AWD standard (instance AWD1): channels monitored: can monitor 1 channel or all channels.groups monitored: ADC groups regular and-or injected.resolution: resolution is not limited (corresponds to ADC resolution configured). AWD flexible (instances AWD2, AWD3): channels monitored: flexible on channels monitored, selection is channel wise, from from 1 to all channels. Specificity of this analog watchdog: Multiple channels can be selected. For example: (LL\_ADC\_AWD\_CHANNEL4\_REG\_INJ | LL\_ADC\_AWD\_CHANNEL5\_REG\_INJ | ...)groups monitored: not selection possible (monitoring on both groups regular and injected). Channels selected are monitored on groups regular and injected: LL\_ADC\_AWD\_CHANNELxx\_REG\_INJ (do not use parameters LL\_ADC\_AWD\_CHANNELxx\_REG and LL\_ADC\_AWD\_CHANNELxx\_INJ)resolution: resolution is limited to 8 bits: if ADC resolution is 12 bits the 4 LSB are ignored, if ADC resolution is 10 bits the 2 LSB are ignored.

- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.

Reference Manual to  
LL API cross  
reference:

- CFGR AWD1CH LL\_ADC\_GetAnalogWDMonitChannels
- CFGR AWD1SGL LL\_ADC\_GetAnalogWDMonitChannels
- CFGR AWD1EN LL\_ADC\_GetAnalogWDMonitChannels
- CFGR JAWD1EN LL\_ADC\_GetAnalogWDMonitChannels
- AWD2CR AWD2CH LL\_ADC\_GetAnalogWDMonitChannels
- AWD3CR AWD3CH LL\_ADC\_GetAnalogWDMonitChannels

## LL\_ADC\_ConfigAnalogWDThresholds

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_ADC_ConfigAnalogWDThresholds(ADC_TypeDef * ADCx, uint32_t AWDDy, uint32_t AWDTresholdHighValue, uint32_t AWDTresholdLowValue)</code>                                                                                                                                                                                                                                                                                         |
| Function description | Set ADC analog watchdog thresholds value of both thresholds high and low.                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>AWDDy:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_AWD1</li> <li>- LL_ADC_AWD2</li> <li>- LL_ADC_AWD3</li> </ul> </li> <li>• <b>AWDTresholdHighValue:</b> Value between Min_Data=0x000 and Max_Data=0xFFFF</li> <li>• <b>AWDTresholdLowValue:</b> Value between Min_Data=0x000 and Max_Data=0xFFFF</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• If value of only one threshold high or low must be set, use function <code>LL_ADC_SetAnalogWDThresholds()</code>.</li> <li>• In case of ADC resolution different of 12 bits, analog watchdog thresholds data require a specific shift. Use helper macro <code>_LL_ADC_ANALOGWD_SET_THRESHOLD_RESOLUTION()</code>.</li> <li>• On this STM32 serie, there are 2 kinds of analog watchdog</li> </ul>                |

instance: AWD standard (instance AWD1): channels monitored: can monitor 1 channel or all channels.groups monitored: ADC groups regular and-or injected.resolution: resolution is not limited (corresponds to ADC resolution configured). AWD flexible (instances AWD2, AWD3): channels monitored: flexible on channels monitored, selection is channel wise, from from 1 to all channels. Specificity of this analog watchdog: Multiple channels can be selected. For example: (LL\_ADC\_AWD\_CHANNEL4\_REG\_INJ | LL\_ADC\_AWD\_CHANNEL5\_REG\_INJ | ...)groups monitored: not selection possible (monitoring on both groups regular and injected). Channels selected are monitored on groups regular and injected: LL\_ADC\_AWD\_CHANNELxx\_REG\_INJ (do not use parameters LL\_ADC\_AWD\_CHANNELxx\_REG and LL\_ADC\_AWD\_CHANNELxx\_INJ)resolution: resolution is limited to 8 bits: if ADC resolution is 12 bits the 4 LSB are ignored, if ADC resolution is 10 bits the 2 LSB are ignored.

- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.

Reference Manual to  
LL API cross  
reference:

- TR1 HT1 LL\_ADC\_ConfigAnalogWDThresholds
- TR2 HT2 LL\_ADC\_ConfigAnalogWDThresholds
- TR3 HT3 LL\_ADC\_ConfigAnalogWDThresholds
- TR1 LT1 LL\_ADC\_ConfigAnalogWDThresholds
- TR2 LT2 LL\_ADC\_ConfigAnalogWDThresholds
- TR3 LT3 LL\_ADC\_ConfigAnalogWDThresholds

## LL\_ADC\_SetAnalogWDThresholds

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_SetAnalogWDThresholds(ADC_TypeDef * ADCx, uint32_t AWDy, uint32_t AWDThresholdsHighLow, uint32_t AWDThresholdValue)</code>                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Set ADC analog watchdog threshold value of threshold high or low.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>AWDy:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_AWD1</li> <li>- LL_ADC_AWD2</li> <li>- LL_ADC_AWD3</li> </ul> </li> <li>• <b>AWDThresholdsHighLow:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_AWD_THRESHOLD_HIGH</li> <li>- LL_ADC_AWD_THRESHOLD_LOW</li> </ul> </li> <li>• <b>AWDThresholdValue:</b> Value between Min_Data=0x000 and Max_Data=0xFFFF</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• If values of both thresholds high or low must be set, use function LL_ADC_ConfigAnalogWDThresholds().</li> <li>• In case of ADC resolution different of 12 bits, analog watchdog thresholds data require a specific shift. Use helper macro<br/><code>__LL_ADC_ANALOGWD_SET_THRESHOLD_RESOLUTION</code></li> </ul>                                                                                                                                                                                                                 |

N().

- On this STM32 serie, there are 2 kinds of analog watchdog instance: AWD standard (instance AWD1): channels monitored: can monitor 1 channel or all channels.groups monitored: ADC groups regular and-or injected.resolution: resolution is not limited (corresponds to ADC resolution configured). AWD flexible (instances AWD2, AWD3): channels monitored: flexible on channels monitored, selection is channel wise, from from 1 to all channels. Specificity of this analog watchdog: Multiple channels can be selected. For example: (LL\_ADC\_AWD\_CHANNEL4\_REG\_INJ | LL\_ADC\_AWD\_CHANNEL5\_REG\_INJ | ...)groups monitored: not selection possible (monitoring on both groups regular and injected). Channels selected are monitored on groups regular and injected: LL\_ADC\_AWD\_CHANNELxx\_REG\_INJ (do not use parameters LL\_ADC\_AWD\_CHANNELxx\_REG and LL\_ADC\_AWD\_CHANNELxx\_INJ)resolution: resolution is limited to 8 bits: if ADC resolution is 12 bits the 4 LSB are ignored, if ADC resolution is 10 bits the 2 LSB are ignored.
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be disabled or enabled without conversion on going on either groups regular or injected.

Reference Manual to  
LL API cross  
reference:

- TR1 HT1 LL\_ADC\_SetAnalogWDThresholds
- TR2 HT2 LL\_ADC\_SetAnalogWDThresholds
- TR3 HT3 LL\_ADC\_SetAnalogWDThresholds
- TR1 LT1 LL\_ADC\_SetAnalogWDThresholds
- TR2 LT2 LL\_ADC\_SetAnalogWDThresholds
- TR3 LT3 LL\_ADC\_SetAnalogWDThresholds

## LL\_ADC\_GetAnalogWDThresholds

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_GetAnalogWDThresholds(ADC_TypeDef * ADCx, uint32_t AWDR, uint32_t AWDRThresholdsHighLow)</code>                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Get ADC analog watchdog threshold value of threshold high, threshold low or raw data with ADC thresholds high and low concatenated.                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>AWDR:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_AWD1</li> <li>- LL_ADC_AWD2</li> <li>- LL_ADC_AWD3</li> </ul> </li> <li>• <b>AWDRThresholdsHighLow:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_AWD_THRESHOLD_HIGH</li> <li>- LL_ADC_AWD_THRESHOLD_LOW</li> <li>- LL_ADC_AWD_THRESHOLDS_HIGH_LOW</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x000 and Max_Data=0xFFFF</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• If raw data with ADC thresholds high and low is retrieved, the data of each threshold high or low can be isolated using helper macro:<br/><code>__LL_ADC_ANALOGWD_THRESHOLDS_HIGH_LOW()</code>.</li> </ul>                                                                                                                                                                                                                                                                              |



- In case of ADC resolution different of 12 bits, analog watchdog thresholds data require a specific shift. Use helper macro `__LL_ADC_ANALOGWD_GET_THRESHOLD_RESOLUTION()`.

Reference Manual to  
LL API cross  
reference:

- TR1 HT1 LL\_ADC\_GetAnalogWDThresholds
- TR2 HT2 LL\_ADC\_GetAnalogWDThresholds
- TR3 HT3 LL\_ADC\_GetAnalogWDThresholds
- TR1 LT1 LL\_ADC\_GetAnalogWDThresholds
- TR2 LT2 LL\_ADC\_GetAnalogWDThresholds
- TR3 LT3 LL\_ADC\_GetAnalogWDThresholds

### **LL\_ADC\_SetMultimode**

Function name

`__STATIC_INLINE void LL_ADC_SetMultimode(ADC_Common_TypeDef * ADCxy_COMMON, uint32_t Multimode)`

Function description

Set ADC multimode configuration to operate in independent mode or multimode (for devices with several ADC instances).

Parameters

- **ADCxy\_COMMON:** ADC common instance (can be set directly from CMSIS definition or by using helper macro `__LL_ADC_COMMON_INSTANCE()`)
- **Multimode:** This parameter can be one of the following values:
  - `LL_ADC_MULTI_INDEPENDENT`
  - `LL_ADC_MULTI_DUAL_REG_SIMULT`
  - `LL_ADC_MULTI_DUAL_REG_INTERL`
  - `LL_ADC_MULTI_DUAL_INJ_SIMULT`
  - `LL_ADC_MULTI_DUAL_INJ_ALTERN`
  - `LL_ADC_MULTI_DUAL_REG_SIM_INJ_SIM`
  - `LL_ADC_MULTI_DUAL_REG_SIM_INJ_ALT`
  - `LL_ADC_MULTI_DUAL_REG_INT_INJ_SIM`

Return values

- **None**

Notes

- If multimode configuration: the selected ADC instance is either master or slave depending on hardware. Refer to reference manual.
- On this STM32 serie, setting of this feature is conditioned to ADC state: All ADC instances of the ADC common group must be disabled. This check can be done with function `LL_ADC_IsEnabled()` for each ADC instance or by using helper macro `__LL_ADC_IS_ENABLED_ALL_COMMON_INSTANCE()`.
- CCR DUAL LL\_ADC\_SetMultimode

Reference Manual to  
LL API cross  
reference:

**LL\_ADC\_GetMultimode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_GetMultimode(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Get ADC multimode configuration to operate in independent mode or multimode (for devices with several ADC instances).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_ADC_MULTI_INDEPENDENT</code></li> <li>- <code>LL_ADC_MULTI_DUAL_REG_SIMULT</code></li> <li>- <code>LL_ADC_MULTI_DUAL_REG_INTERL</code></li> <li>- <code>LL_ADC_MULTI_DUAL_INJ_SIMULT</code></li> <li>- <code>LL_ADC_MULTI_DUAL_INJ_ALTERN</code></li> <li>- <code>LL_ADC_MULTI_DUAL_REG_SIM_INJ_SIM</code></li> <li>- <code>LL_ADC_MULTI_DUAL_REG_SIM_INJ_ALT</code></li> <li>- <code>LL_ADC_MULTI_DUAL_REG_INT_INJ_SIM</code></li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• If multimode configuration: the selected ADC instance is either master or slave depending on hardware. Refer to reference manual.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR DUAL LL_ADC_SetMultimode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**LL\_ADC\_SetMultiDMATransfer**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_SetMultiDMATransfer(ADC_Common_TypeDef * ADCxy_COMMON, uint32_t MultiDMATransfer)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Set ADC multimode conversion data transfer: no transfer or transfer by DMA.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> <li>• <b>MultiDMATransfer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_ADC_MULTI_REG_DMA_EACH_ADC</code></li> <li>- <code>LL_ADC_MULTI_REG_DMA_LIMIT_RES12_10B</code></li> <li>- <code>LL_ADC_MULTI_REG_DMA_LIMIT_RES8_6B</code></li> <li>- <code>LL_ADC_MULTI_REG_DMA_UNLMT_RES12_10B</code></li> <li>- <code>LL_ADC_MULTI_REG_DMA_UNLMT_RES8_6B</code></li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Notes                | <ul style="list-style-type: none"> <li>• If ADC multimode transfer by DMA is not selected: each ADC uses its own DMA channel, with its individual DMA transfer settings. If ADC multimode transfer by DMA is selected: One DMA channel is used for both ADC (DMA of ADC master) Specifies the DMA requests mode: Limited mode (One shot mode): DMA transfer requests are stopped when number of DMA data transfers (number of ADC conversions) is reached.</li> </ul>                                                                                                                                                                            |

This ADC mode is intended to be used with DMA mode non-circular.Unlimited mode: DMA transfer requests are unlimited, whatever number of DMA data transfers (number of ADC conversions). This ADC mode is intended to be used with DMA mode circular.

- If ADC DMA requests mode is set to unlimited and DMA is set to mode non-circular: when DMA transfers size will be reached, DMA will stop transfers of ADC conversions data ADC will raise an overrun error (overrun flag and interruption if enabled).
- How to retrieve multimode conversion data: Whatever multimode transfer by DMA setting: using function LL\_ADC\_REG\_ReadMultiConversionData32(). If ADC multimode transfer by DMA is selected: conversion data is a raw data with ADC master and slave concatenated. A macro is available to get the conversion data of ADC master or ADC slave: see helper macro \_\_LL\_ADC\_MULTI\_CONV\_DATA\_MASTER\_SLAVE().
- On this STM32 serie, setting of this feature is conditioned to ADC state: All ADC instances of the ADC common group must be disabled or enabled without conversion on going on group regular.

Reference Manual to  
LL API cross  
reference:

- CCR MDMA LL\_ADC\_SetMultiDMATransfer
- CCR DMACFG LL\_ADC\_SetMultiDMATransfer

### **LL\_ADC\_GetMultiDMATransfer**

Function name **STATIC\_INLINE uint32\_t LL\_ADC\_GetMultiDMATransfer(ADC\_Common\_TypeDef \* ADCxy\_COMMON)**

Function description Get ADC multimode conversion data transfer: no transfer or transfer by DMA.

Parameters **ADCxy\_COMMON:** ADC common instance (can be set directly from CMSIS definition or by using helper macro \_\_LL\_ADC\_COMMON\_INSTANCE() )

Return values **Returned:** value can be one of the following values:

- LL\_ADC\_MULTI\_REG\_DMA\_EACH\_ADC
- LL\_ADC\_MULTI\_REG\_DMA\_LIMIT\_RES12\_10B
- LL\_ADC\_MULTI\_REG\_DMA\_LIMIT\_RES8\_6B
- LL\_ADC\_MULTI\_REG\_DMA\_UNLMT\_RES12\_10B
- LL\_ADC\_MULTI\_REG\_DMA\_UNLMT\_RES8\_6B

Notes • If ADC multimode transfer by DMA is not selected: each ADC uses its own DMA channel, with its individual DMA transfer settings. If ADC multimode transfer by DMA is selected: One DMA channel is used for both ADC (DMA of ADC master) Specifies the DMA requests mode: Limited mode (One shot mode): DMA transfer requests are stopped when number of DMA data transfers (number of ADC conversions) is reached. This ADC mode is intended to be used with DMA mode non-circular.Unlimited mode: DMA transfer requests are unlimited, whatever number of DMA data transfers (number of ADC conversions). This ADC mode is intended to be used with

- DMA mode circular.
- If ADC DMA requests mode is set to unlimited and DMA is set to mode non-circular: when DMA transfers size will be reached, DMA will stop transfers of ADC conversions data ADC will raise an overrun error (overrun flag and interruption if enabled).
- How to retrieve multimode conversion data: Whatever multimode transfer by DMA setting: using function LL\_ADC\_REG\_ReadMultiConversionData32(). If ADC multimode transfer by DMA is selected: conversion data is a raw data with ADC master and slave concatenated. A macro is available to get the conversion data of ADC master or ADC slave: see helper macro \_\_LL\_ADC\_MULTI\_CONV\_DATA\_MASTER\_SLAVE().

Reference Manual to  
LL API cross  
reference:

- CCR MDMA LL\_ADC\_GetMultiDMATransfer
- CCR DMACFG LL\_ADC\_GetMultiDMATransfer

### **LL\_ADC\_SetMultiTwoSamplingDelay**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_SetMultiTwoSamplingDelay(ADC_Common_TypeDef * ADCxy_COMMON, uint32_t MultiTwoSamplingDelay)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description | Set ADC multimode delay between 2 sampling phases.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro __LL_ADC_COMMON_INSTANCE() )</li> <li><b>MultiTwoSamplingDelay:</b> This parameter can be one of the following values: (1) Parameter available only if ADC resolution is 12, 10 or 8 bits. <ul style="list-style-type: none"> <li>- LL_ADC_MULTI_TWOSMP_DELAY_1CYCLE</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_2CYCLES</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_3CYCLES</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_4CYCLES</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_5CYCLES</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_6CYCLES (1)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_7CYCLES (1)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_8CYCLES (2)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_9CYCLES (2)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_10CYCLES (2)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_11CYCLES (3)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_12CYCLES (3)</li> </ul> </li> <li>(2) Parameter available only if ADC resolution is 12 or 10 bits.</li> <li>(3) Parameter available only if ADC resolution is 12 bits.</li> </ul> <li><b>None</b></li> |
| Return values        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Notes                | <ul style="list-style-type: none"> <li>The sampling delay range depends on ADC resolution: ADC resolution 12 bits can have maximum delay of 12 cycles. ADC resolution 10 bits can have maximum delay of 10 cycles. ADC resolution 8 bits can have maximum delay of 8 cycles. ADC resolution 6 bits can have maximum delay of 6 cycles.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

Reference Manual to  
LL API cross  
reference:

- On this STM32 serie, setting of this feature is conditioned to ADC state: All ADC instances of the ADC common group must be disabled. This check can be done with function LL\_ADC\_IsEnabled() for each ADC instance or by using helper macro helper macro \_\_LL\_ADC\_IS\_ENABLED\_ALL\_COMMON\_INSTANCE().
- CCR DELAY LL\_ADC\_SetMultiTwoSamplingDelay

### **LL\_ADC\_GetMultiTwoSamplingDelay**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_GetMultiTwoSamplingDelay<br/>(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description                              | Get ADC multimode delay between 2 sampling phases.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro __LL_ADC_COMMON_INSTANCE() )</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (1) Parameter available only if ADC resolution is 12, 10 or 8 bits. <ul style="list-style-type: none"> <li>- LL_ADC_MULTI_TWOSMP_DELAY_1CYCLE</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_2CYCLES</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_3CYCLES</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_4CYCLES</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_5CYCLES</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_6CYCLES (1)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_7CYCLES (1)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_8CYCLES (2)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_9CYCLES (2)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_10CYCLES (2)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_11CYCLES (3)</li> <li>- LL_ADC_MULTI_TWOSMP_DELAY_12CYCLES (3)</li> </ul> </li> <li>• (2) Parameter available only if ADC resolution is 12 or 10 bits.</li> <li>• (3) Parameter available only if ADC resolution is 12 bits.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR DELAY LL_ADC_GetMultiTwoSamplingDelay</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

### **LL\_ADC\_EnableInternalRegulator**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_EnableInternalRegulator<br/>(ADC_TypeDef * ADCx)</code>                                                                                      |
| Function description | Enable ADC instance internal voltage regulator.                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                |
| Notes                | <ul style="list-style-type: none"> <li>• On this STM32 serie, after ADC internal voltage regulator enable, a delay for ADC internal voltage regulator stabilization</li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Reference Manual to<br>LL API cross<br>reference: | is required before performing a ADC calibration or ADC enable. Refer to device datasheet, parameter tADCVREG_STUP. Refer to literal LL_ADC_DELAY_INTERNAL_REGUL_STAB_US.<br><ul style="list-style-type: none"> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be ADC disabled.</li> <li>• CR ADVREGEN LL_ADC_EnableInternalRegulator</li> </ul> |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### LL\_ADC\_DisableInternalRegulator

|                                                                                                                                                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|--------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name<br><br>Function description<br><br>Parameters<br><br>Return values<br><br>Notes<br><br>Reference Manual to<br>LL API cross<br>reference: | <b>_STATIC_INLINE void LL_ADC_DisableInternalRegulator (ADC_TypeDef * ADCx)</b><br>Disable ADC internal voltage regulator.<br><ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>None</b></li> </ul> <ul style="list-style-type: none"> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be ADC disabled.</li> <li>• CR ADVREGEN LL_ADC_DisableInternalRegulator</li> </ul> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### LL\_ADC\_IsInternalRegulatorEnabled

|                                                                                                                                                        |                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name<br><br>Function description<br><br>Parameters<br><br>Return values<br><br>Notes<br><br>Reference Manual to<br>LL API cross<br>reference: | <b>_STATIC_INLINE uint32_t LL_ADC_IsInternalRegulatorEnabled (ADC_TypeDef * ADCx)</b><br>Get the selected ADC instance internal voltage regulator state.<br><ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>0:</b> internal regulator is disabled, 1: internal regulator is enabled.</li> </ul> <ul style="list-style-type: none"> <li>• CR ADVREGEN LL_ADC_IsInternalRegulatorEnabled</li> </ul> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### LL\_ADC\_Enable

|                                                                                                                                                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|--------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name<br><br>Function description<br><br>Parameters<br><br>Return values<br><br>Notes<br><br>Reference Manual to<br>LL API cross<br>reference: | <b>_STATIC_INLINE void LL_ADC_Enable (ADC_TypeDef * ADCx)</b><br>Enable the selected ADC instance.<br><ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>None</b></li> </ul> <ul style="list-style-type: none"> <li>• On this STM32 serie, after ADC enable, a delay for ADC internal analog stabilization is required before performing a ADC conversion start. Refer to device datasheet, parameter tSTAB.</li> <li>• On this STM32 serie, flag LL_ADC_FLAG_ADRDY is raised</li> </ul> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Reference Manual to  
LL API cross  
reference:

when the ADC is enabled and when conversion clock is active. (not only core clock: this ADC has a dual clock domain)

- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be ADC disabled and ADC internal voltage regulator enabled.

- CR ADEN LL\_ADC\_Enable

### **LL\_ADC\_Disable**

|                                                   |                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_Disable (ADC_TypeDef * ADCx)</code>                                                                                                                                                                     |
| Function description                              | Disable the selected ADC instance.                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be not disabled. Must be enabled without conversion on going on either groups regular or injected.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR ADDIS LL_ADC_Disable</li> </ul>                                                                                                                                                               |

### **LL\_ADC\_IsEnabled**

|                                                   |                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_IsEnabled (ADC_TypeDef * ADCx)</code>                                                                                                                                                    |
| Function description                              | Get the selected ADC instance enable state.                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>0:</b> ADC is disabled, 1: ADC is enabled.</li> </ul>                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• On this STM32 serie, flag LL_ADC_FLAG_ADRDY is raised when the ADC is enabled and when conversion clock is active. (not only core clock: this ADC has a dual clock domain)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR ADEN LL_ADC_IsEnabled</li> </ul>                                                                                                                                                   |

### **LL\_ADC\_IsDisableOngoing**

|                      |                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_IsDisableOngoing (ADC_TypeDef * ADCx)</code>             |
| Function description | Get the selected ADC instance disable state.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>0:</b> no ADC disable command on going.</li> </ul> |

Reference Manual to  
LL API cross  
reference:

- CR ADDIS LL\_ADC\_IsDisableOngoing

### LL\_ADC\_StartCalibration

Function name

`_STATIC_INLINE void LL_ADC_StartCalibration  
(ADC_TypeDef * ADCx, uint32_t SingleDiff)`

Function description

Start ADC calibration in the mode single-ended or differential (for devices with differential mode available).

Parameters

- **ADCx:** ADC instance
- **SingleDiff:** This parameter can be one of the following values:
  - LL\_ADC\_SINGLE\_ENDED
  - LL\_ADC\_DIFFERENTIAL\_ENDED

Return values

- **None**

Notes

- On this STM32 serie, a minimum number of ADC clock cycles are required between ADC end of calibration and ADC enable. Refer to literal `LL_ADC_DELAY_CALIB_ENABLE_ADC_CYCLES`.
- For devices with differential mode available: Calibration of offset is specific to each of single-ended and differential modes (calibration run must be performed for each of these differential modes, if used afterwards and if the application requires their calibration).
- On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be ADC disabled.

Reference Manual to  
LL API cross  
reference:

- CR ADCAL LL\_ADC\_StartCalibration
- CR ADCALDIF LL\_ADC\_StartCalibration

### LL\_ADC\_IsCalibrationOnGoing

Function name

`_STATIC_INLINE uint32_t LL_ADC_IsCalibrationOnGoing  
(ADC_TypeDef * ADCx)`

Function description

Get ADC calibration state.

Parameters

- **ADCx:** ADC instance

Return values

- **0:** calibration complete, 1: calibration in progress.

Reference Manual to  
LL API cross  
reference:

- CR ADCAL LL\_ADC\_IsCalibrationOnGoing

### LL\_ADC\_REG\_StartConversion

Function name

`_STATIC_INLINE void LL_ADC_REG_StartConversion  
(ADC_TypeDef * ADCx)`

Function description

Start ADC group regular conversion.

Parameters

- **ADCx:** ADC instance

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>On this STM32 serie, this function is relevant for both internal trigger (SW start) and external trigger: If ADC trigger has been set to software start, ADC conversion starts immediately. If ADC trigger has been set to external trigger, ADC conversion will start at next trigger event (on the selected trigger edge) following the ADC start conversion command.</li> <li>On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be enabled without conversion on going on group regular, without conversion stop command on going on group regular, without ADC disable command on going.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR ADSTART LL_ADC_REG_StartConversion</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

### LL\_ADC\_REG\_StopConversion

|                                                   |                                                                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_ADC_REG_StopConversion(ADC_TypeDef * ADCx)</code></b>                                                                                                                                            |
| Function description                              | Stop ADC group regular conversion.                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be enabled with conversion on going on group regular, without ADC disable command on going.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR ADSTP LL_ADC_REG_StopConversion</li> </ul>                                                                                                                                             |

### LL\_ADC\_REG\_IsConversionOngoing

|                      |                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_ADC_REG_IsConversionOngoing(ADC_TypeDef * ADCx)</code></b>              |
| Function description | Get ADC group regular conversion state.                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                 |
| Return values        | <ul style="list-style-type: none"> <li><b>0:</b> no conversion is on going on ADC group regular.</li> </ul> |

Reference Manual to  
LL API cross  
reference:  
[CR ADSTART LL\\_ADC\\_REG\\_IsConversionOngoing](#)

### LL\_ADC\_REG\_IsStopConversionOngoing

|                      |                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_ADC_REG_IsStopConversionOngoing(ADC_TypeDef * ADCx)</code></b> |
| Function description | Get ADC group regular command of conversion stop state.                                            |

|                                                   |                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li><b>0:</b> no command of conversion stop is on going on ADC group regular.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR ADSTP LL_ADC_REG_IsStopConversionOngoing</li> </ul>                               |

### LL\_ADC\_REG\_ReadConversionData32

|                                                   |                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_REG_ReadConversionData32 (ADC_TypeDef *<br/>ADCx)</code>                                                                                  |
| Function description                              | Get ADC group regular conversion data, range fit for all ADC configurations: all ADC resolutions and all oversampling increased data width (for devices with feature oversampling). |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x00000000 and Max_Data=0xFFFFFFFF</li> </ul>                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>DR RDATA LL_ADC_REG_ReadConversionData32</li> </ul>                                                                                          |

### LL\_ADC\_REG\_ReadConversionData12

|                                                   |                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint16_t<br/>LL_ADC_REG_ReadConversionData12 (ADC_TypeDef *<br/>ADCx)</code>                                                                                                         |
| Function description                              | Get ADC group regular conversion data, range fit for ADC resolution 12 bits.                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x000 and Max_Data=0xFFFF</li> </ul>                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>For devices with feature oversampling: Oversampling can increase data width, function for extended range may be needed: LL_ADC_REG_ReadConversionData32.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>DR RDATA LL_ADC_REG_ReadConversionData12</li> </ul>                                                                                                                 |

### LL\_ADC\_REG\_ReadConversionData10

|                      |                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint16_t<br/>LL_ADC_REG_ReadConversionData10 (ADC_TypeDef *<br/>ADCx)</code>        |
| Function description | Get ADC group regular conversion data, range fit for ADC resolution 10 bits.                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                               |
| Return values        | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x000 and Max_Data=0x3FF</li> </ul> |

- Notes**
- For devices with feature oversampling: Oversampling can increase data width, function for extended range may be needed: LL\_ADC\_REG\_ReadConversionData32.

- Reference Manual to LL API cross reference:**
- DR RDATA LL\_ADC\_REG\_ReadConversionData10

### LL\_ADC\_REG\_ReadConversionData8

|                                                    |                                                                                                                                                                                                            |
|----------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>                               | <code>__STATIC_INLINE uint8_t<br/>LL_ADC_REG_ReadConversionData8 (ADC_TypeDef * ADCx)</code>                                                                                                               |
| <b>Function description</b>                        | Get ADC group regular conversion data, range fit for ADC resolution 8 bits.                                                                                                                                |
| <b>Parameters</b>                                  | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                |
| <b>Return values</b>                               | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x00 and Max_Data=0xFF</li> </ul>                                                                                                    |
| <b>Notes</b>                                       | <ul style="list-style-type: none"> <li>For devices with feature oversampling: Oversampling can increase data width, function for extended range may be needed: LL_ADC_REG_ReadConversionData32.</li> </ul> |
| <b>Reference Manual to LL API cross reference:</b> | <ul style="list-style-type: none"> <li>DR RDATA LL_ADC_REG_ReadConversionData8</li> </ul>                                                                                                                  |

### LL\_ADC\_REG\_ReadConversionData6

|                                                    |                                                                                                                                                                                                            |
|----------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>                               | <code>__STATIC_INLINE uint8_t<br/>LL_ADC_REG_ReadConversionData6 (ADC_TypeDef * ADCx)</code>                                                                                                               |
| <b>Function description</b>                        | Get ADC group regular conversion data, range fit for ADC resolution 6 bits.                                                                                                                                |
| <b>Parameters</b>                                  | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                |
| <b>Return values</b>                               | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x00 and Max_Data=0x3F</li> </ul>                                                                                                    |
| <b>Notes</b>                                       | <ul style="list-style-type: none"> <li>For devices with feature oversampling: Oversampling can increase data width, function for extended range may be needed: LL_ADC_REG_ReadConversionData32.</li> </ul> |
| <b>Reference Manual to LL API cross reference:</b> | <ul style="list-style-type: none"> <li>DR RDATA LL_ADC_REG_ReadConversionData6</li> </ul>                                                                                                                  |

### LL\_ADC\_REG\_ReadMultiConversionData32

|                             |                                                                                                                                                                                                         |
|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>        | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_REG_ReadMultiConversionData32<br/>(ADC_Common_TypeDef * ADCxy_COMMON, uint32_t<br/>ConversionData)</code>                                                     |
| <b>Function description</b> | Get ADC multimode conversion data of ADC master, ADC slave or raw data with ADC master and slave concatenated.                                                                                          |
| <b>Parameters</b>           | <ul style="list-style-type: none"> <li><b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                             | <ul style="list-style-type: none"> <li>• <b>ConversionData:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_ADC_MULTI_MASTER</li> <li>– LL_ADC_MULTI_SLAVE</li> <li>– LL_ADC_MULTI_MASTER_SLAVE</li> </ul> </li> </ul>                                                                                                                                                                              |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00000000 and Max_Data=0xFFFFFFFF</li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Notes                                       | <ul style="list-style-type: none"> <li>• If raw data with ADC master and slave concatenated is retrieved, a macro is available to get the conversion data of ADC master or ADC slave: see helper macro <code>__LL_ADC_MULTI_CONV_DATA_MASTER_SLAVE()</code>. (however this macro is mainly intended for multimode transfer by DMA, because this function can do the same by getting multimode conversion data of ADC master or ADC slave separately).</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CDR RDATA_MST<br/><code>LL_ADC_REG_ReadMultiConversionData32</code></li> <li>• CDR RDATA_SLV<br/><code>LL_ADC_REG_ReadMultiConversionData32</code></li> </ul>                                                                                                                                                                                                                                                           |

### LL\_ADC\_INJ\_StartConversion

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_ADC_INJ_StartConversion(ADC_TypeDef * ADCx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                        | Start ADC group injected conversion.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                                       | <ul style="list-style-type: none"> <li>• On this STM32 serie, this function is relevant for both internal trigger (SW start) and external trigger: If ADC trigger has been set to software start, ADC conversion starts immediately. If ADC trigger has been set to external trigger, ADC conversion will start at next trigger event (on the selected trigger edge) following the ADC start conversion command.</li> <li>• On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be enabled without conversion on going on group injected, without conversion stop command on going on group injected, without ADC disable command on going.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR JADSTART LL_ADC_INJ_StartConversion</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

### LL\_ADC\_INJ\_StopConversion

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_INJ_StopConversion(ADC_TypeDef * ADCx)</code> |
| Function description | Stop ADC group injected conversion.                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>   |

|                                             |                                                                                                                                                                                                                                   |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                     |
| Notes                                       | <ul style="list-style-type: none"> <li>On this STM32 serie, setting of this feature is conditioned to ADC state: ADC must be enabled with conversion on going on group injected, without ADC disable command on going.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR JADSTP LL_ADC_INJ_StopConversion</li> </ul>                                                                                                                                             |

### LL\_ADC\_INJ\_IsConversionOngoing

|                                             |                                                                                                              |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_INJ_IsConversionOngoing (ADC_TypeDef * ADCx)</code>                |
| Function description                        | Get ADC group injected conversion state.                                                                     |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                  |
| Return values                               | <ul style="list-style-type: none"> <li><b>0:</b> no conversion is on going on ADC group injected.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR JADSTART LL_ADC_INJ_IsConversionOngoing</li> </ul>                 |

### LL\_ADC\_INJ\_IsStopConversionOngoing

|                                             |                                                                                                                              |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_INJ_IsStopConversionOngoing (ADC_TypeDef * ADCx)</code>                            |
| Function description                        | Get ADC group injected command of conversion stop state.                                                                     |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                  |
| Return values                               | <ul style="list-style-type: none"> <li><b>0:</b> no command of conversion stop is on going on ADC group injected.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR JADSTP LL_ADC_INJ_IsStopConversionOngoing</li> </ul>                               |

### LL\_ADC\_INJ\_ReadConversionData32

|                      |                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_INJ_ReadConversionData32 (ADC_TypeDef * ADCx,<br/>uint32_t Rank)</code>                                                                                                                                                                                                                                                      |
| Function description | Get ADC group regular conversion data, range fit for all ADC configurations: all ADC resolutions and all oversampling increased data width (for devices with feature oversampling).                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> <li><b>Rank:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_ADC_INJ_RANK_1</code></li> <li>– <code>LL_ADC_INJ_RANK_2</code></li> <li>– <code>LL_ADC_INJ_RANK_3</code></li> <li>– <code>LL_ADC_INJ_RANK_4</code></li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>Value:</b> between <code>Min_Data=0x00000000</code> and</li> </ul>                                                                                                                                                                                                                                                           |

Max\_Data=0xFFFFFFFF

Reference Manual to  
LL API cross  
reference:

- JDR1 JDATA LL\_ADC\_INJ\_ReadConversionData32
- JDR2 JDATA LL\_ADC\_INJ\_ReadConversionData32
- JDR3 JDATA LL\_ADC\_INJ\_ReadConversionData32
- JDR4 JDATA LL\_ADC\_INJ\_ReadConversionData32

### **LL\_ADC\_INJ\_ReadConversionData12**

|                                                   |                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint16_t<br/>LL_ADC_INJ_ReadConversionData12 (ADC_TypeDef * ADCx,<br/>uint32_t Rank)</code>                                                                                                                                                                                                      |
| Function description                              | Get ADC group injected conversion data, range fit for ADC resolution 12 bits.                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Rank:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_ADC_INJ_RANK_1</li> <li>– LL_ADC_INJ_RANK_2</li> <li>– LL_ADC_INJ_RANK_3</li> <li>– LL_ADC_INJ_RANK_4</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x000 and Max_Data=0xFFF</li> </ul>                                                                                                                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• For devices with feature oversampling: Oversampling can increase data width, function for extended range may be needed: LL_ADC_INJ_ReadConversionData32.</li> </ul>                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• JDR1 JDATA LL_ADC_INJ_ReadConversionData12</li> <li>• JDR2 JDATA LL_ADC_INJ_ReadConversionData12</li> <li>• JDR3 JDATA LL_ADC_INJ_ReadConversionData12</li> <li>• JDR4 JDATA LL_ADC_INJ_ReadConversionData12</li> </ul>                                                       |

### **LL\_ADC\_INJ\_ReadConversionData10**

|                                     |                                                                                                                                                                                                                                                                                                                        |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                       | <code>__STATIC_INLINE uint16_t<br/>LL_ADC_INJ_ReadConversionData10 (ADC_TypeDef * ADCx,<br/>uint32_t Rank)</code>                                                                                                                                                                                                      |
| Function description                | Get ADC group injected conversion data, range fit for ADC resolution 10 bits.                                                                                                                                                                                                                                          |
| Parameters                          | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Rank:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_ADC_INJ_RANK_1</li> <li>– LL_ADC_INJ_RANK_2</li> <li>– LL_ADC_INJ_RANK_3</li> <li>– LL_ADC_INJ_RANK_4</li> </ul> </li> </ul> |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x000 and Max_Data=0x3FF</li> </ul>                                                                                                                                                                                                            |
| Notes                               | <ul style="list-style-type: none"> <li>• For devices with feature oversampling: Oversampling can increase data width, function for extended range may be needed: LL_ADC_INJ_ReadConversionData32.</li> </ul>                                                                                                           |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• JDR1 JDATA LL_ADC_INJ_ReadConversionData10</li> <li>• JDR2 JDATA LL_ADC_INJ_ReadConversionData10</li> <li>• JDR3 JDATA LL_ADC_INJ_ReadConversionData10</li> </ul>                                                                                                             |

- JDR4 JDATA LL\_ADC\_INJ\_ReadConversionData10

### **LL\_ADC\_INJ\_ReadConversionData8**

|                                                   |                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><u>__STATIC_INLINE uint8_t<br/>LL_ADC_INJ_ReadConversionData8 (ADC_TypeDef * ADCx,<br/>uint32_t Rank)</u></b>                                                                                                                                                                                                       |
| Function description                              | Get ADC group injected conversion data, range fit for ADC resolution 8 bits.                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Rank:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_INJ_RANK_1</li> <li>- LL_ADC_INJ_RANK_2</li> <li>- LL_ADC_INJ_RANK_3</li> <li>- LL_ADC_INJ_RANK_4</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xFF</li> </ul>                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• For devices with feature oversampling: Oversampling can increase data width, function for extended range may be needed: LL_ADC_INJ_ReadConversionData32.</li> </ul>                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• JDR1 JDATA LL_ADC_INJ_ReadConversionData8</li> <li>• JDR2 JDATA LL_ADC_INJ_ReadConversionData8</li> <li>• JDR3 JDATA LL_ADC_INJ_ReadConversionData8</li> <li>• JDR4 JDATA LL_ADC_INJ_ReadConversionData8</li> </ul>                                                           |

### **LL\_ADC\_INJ\_ReadConversionData6**

|                                                   |                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><u>__STATIC_INLINE uint8_t<br/>LL_ADC_INJ_ReadConversionData6 (ADC_TypeDef * ADCx,<br/>uint32_t Rank)</u></b>                                                                                                                                                                                                       |
| Function description                              | Get ADC group injected conversion data, range fit for ADC resolution 6 bits.                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>Rank:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_ADC_INJ_RANK_1</li> <li>- LL_ADC_INJ_RANK_2</li> <li>- LL_ADC_INJ_RANK_3</li> <li>- LL_ADC_INJ_RANK_4</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x3F</li> </ul>                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• For devices with feature oversampling: Oversampling can increase data width, function for extended range may be needed: LL_ADC_INJ_ReadConversionData32.</li> </ul>                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• JDR1 JDATA LL_ADC_INJ_ReadConversionData6</li> <li>• JDR2 JDATA LL_ADC_INJ_ReadConversionData6</li> <li>• JDR3 JDATA LL_ADC_INJ_ReadConversionData6</li> <li>• JDR4 JDATA LL_ADC_INJ_ReadConversionData6</li> </ul>                                                           |

**LL\_ADC\_IsActiveFlag\_ADRDY**

|                                                   |                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_ADRDY(ADC_TypeDef * ADCx)</code>                                                                                                                                             |
| Function description                              | Get flag ADC ready.                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• On this STM32 serie, flag LL_ADC_FLAG_ADRDY is raised when the ADC is enabled and when conversion clock is active. (not only core clock: this ADC has a dual clock domain)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR ADRDY LL_ADC_IsActiveFlag_ADRDY</li> </ul>                                                                                                                                        |

**LL\_ADC\_IsActiveFlag\_EOC**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_EOC(ADC_TypeDef * ADCx)</code>    |
| Function description                              | Get flag ADC group regular end of unitary conversion.                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR EOC LL_ADC_IsActiveFlag_EOC</li> </ul> |

**LL\_ADC\_IsActiveFlag\_EOS**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_EOS(ADC_TypeDef * ADCx)</code>    |
| Function description                              | Get flag ADC group regular end of sequence conversions.                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR EOS LL_ADC_IsActiveFlag_EOS</li> </ul> |

**LL\_ADC\_IsActiveFlag\_OVR**

|                      |                                                                                    |
|----------------------|------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_OVR(ADC_TypeDef * ADCx)</code>   |
| Function description | Get flag ADC group regular overrun.                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |

Reference Manual to  
LL API cross  
reference:

- ISR OVR LL\_ADC\_IsActiveFlag\_OVR

### **LL\_ADC\_IsActiveFlag\_EOSMP**

Function name            **\_STATIC\_INLINE uint32\_t LL\_ADC\_IsActiveFlag\_EOSMP(ADC\_TypeDef \* ADCx)**

Function description     Get flag ADC group regular end of sampling phase.

Parameters              • **ADCx:** ADC instance

Return values           • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR EOSMP LL\_ADC\_IsActiveFlag\_EOSMP

### **LL\_ADC\_IsActiveFlag\_JEOC**

Function name            **\_STATIC\_INLINE uint32\_t LL\_ADC\_IsActiveFlag\_JEOC(ADC\_TypeDef \* ADCx)**

Function description     Get flag ADC group injected end of unitary conversion.

Parameters              • **ADCx:** ADC instance

Return values           • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR JEOC LL\_ADC\_IsActiveFlag\_JEOC

### **LL\_ADC\_IsActiveFlag\_JEOS**

Function name            **\_STATIC\_INLINE uint32\_t LL\_ADC\_IsActiveFlag\_JEOS(ADC\_TypeDef \* ADCx)**

Function description     Get flag ADC group injected end of sequence conversions.

Parameters              • **ADCx:** ADC instance

Return values           • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR JEOS LL\_ADC\_IsActiveFlag\_JEOS

### **LL\_ADC\_IsActiveFlag\_JQOVF**

Function name            **\_STATIC\_INLINE uint32\_t LL\_ADC\_IsActiveFlag\_JQOVF(ADC\_TypeDef \* ADCx)**

Function description     Get flag ADC group injected contexts queue overflow.

Parameters              • **ADCx:** ADC instance

Return values           • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR JQOVF LL\_ADC\_IsActiveFlag\_JQOVF

### **LL\_ADC\_IsActiveFlag\_AWD1**

Function name      **`_STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_AWD1(ADC_TypeDef * ADCx)`**

Function description      Get flag ADC analog watchdog 1 flag.

Parameters      • **ADCx:** ADC instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR AWD1 LL\_ADC\_IsActiveFlag\_AWD1

### **LL\_ADC\_IsActiveFlag\_AWD2**

Function name      **`_STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_AWD2(ADC_TypeDef * ADCx)`**

Function description      Get flag ADC analog watchdog 2.

Parameters      • **ADCx:** ADC instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR AWD2 LL\_ADC\_IsActiveFlag\_AWD2

### **LL\_ADC\_IsActiveFlag\_AWD3**

Function name      **`_STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_AWD3(ADC_TypeDef * ADCx)`**

Function description      Get flag ADC analog watchdog 3.

Parameters      • **ADCx:** ADC instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR AWD3 LL\_ADC\_IsActiveFlag\_AWD3

### **LL\_ADC\_ClearFlag\_ADRDY**

Function name      **`_STATIC_INLINE void LL_ADC_ClearFlag_ADRDY(ADC_TypeDef * ADCx)`**

Function description      Clear flag ADC ready.

Parameters      • **ADCx:** ADC instance

Return values      • **None**

Notes      • On this STM32 serie, flag LL\_ADC\_FLAG\_ADRDY is raised when the ADC is enabled and when conversion clock is

active. (not only core clock: this ADC has a dual clock domain)

Reference Manual to  
LL API cross  
reference:

- ISR ADRDY LL\_ADC\_ClearFlag\_ADRDY

### **LL\_ADC\_ClearFlag\_EOC**

Function name      **\_STATIC\_INLINE void LL\_ADC\_ClearFlag\_EOC(ADC\_TypeDef \* ADCx)**

Function description      Clear flag ADC group regular end of unitary conversion.

Parameters      • **ADCx:** ADC instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:

- ISR EOC LL\_ADC\_ClearFlag\_EOC

### **LL\_ADC\_ClearFlag\_EOS**

Function name      **\_STATIC\_INLINE void LL\_ADC\_ClearFlag\_EOS(ADC\_TypeDef \* ADCx)**

Function description      Clear flag ADC group regular end of sequence conversions.

Parameters      • **ADCx:** ADC instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:

- ISR EOS LL\_ADC\_ClearFlag\_EOS

### **LL\_ADC\_ClearFlag\_OVR**

Function name      **\_STATIC\_INLINE void LL\_ADC\_ClearFlag\_OVR(ADC\_TypeDef \* ADCx)**

Function description      Clear flag ADC group regular overrun.

Parameters      • **ADCx:** ADC instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:

- ISR OVR LL\_ADC\_ClearFlag\_OVR

### **LL\_ADC\_ClearFlag\_EOSMP**

Function name      **\_STATIC\_INLINE void LL\_ADC\_ClearFlag\_EOSMP(ADC\_TypeDef \* ADCx)**

Function description      Clear flag ADC group regular end of sampling phase.

Parameters      • **ADCx:** ADC instance

---

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR EOSMP LL_ADC_ClearFlag_EOSMP</li> </ul> |

### LL\_ADC\_ClearFlag\_JEOC

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_ADC_ClearFlag_JEOC(ADC_TypeDef * ADCx)</code>       |
| Function description                              | Clear flag ADC group injected end of unitary conversion.                         |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR JEOC LL_ADC_ClearFlag_JEOC</li> </ul> |

### LL\_ADC\_ClearFlag\_JEOS

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_ADC_ClearFlag_JEOS(ADC_TypeDef * ADCx)</code>       |
| Function description                              | Clear flag ADC group injected end of sequence conversions.                       |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR JEOS LL_ADC_ClearFlag_JEOS</li> </ul> |

### LL\_ADC\_ClearFlag\_JQOVF

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_ADC_ClearFlag_JQOVF(ADC_TypeDef * ADCx)</code>        |
| Function description                              | Clear flag ADC group injected contexts queue overflow.                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR JQOVF LL_ADC_ClearFlag_JQOVF</li> </ul> |

### LL\_ADC\_ClearFlag\_AWD1

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_ADC_ClearFlag_AWD1(ADC_TypeDef * ADCx)</code>  |
| Function description | Clear flag ADC analog watchdog 1.                                           |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>               |

- Reference Manual to  
LL API cross  
reference:
- ISR AWD1 LL\_ADC\_ClearFlag\_AWD1

### LL\_ADC\_ClearFlag\_AWD2

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_ClearFlag_AWD2(ADC_TypeDef * ADCx)</code> |
| Function description | Clear flag ADC analog watchdog 2.                                           |
| Parameters           | <ul style="list-style-type: none"><li>• <b>ADCx:</b> ADC instance</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>               |

Reference Manual to  
LL API cross  
reference:

- ISR AWD2 LL\_ADC\_ClearFlag\_AWD2

### LL\_ADC\_ClearFlag\_AWD3

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_ADC_ClearFlag_AWD3(ADC_TypeDef * ADCx)</code> |
| Function description | Clear flag ADC analog watchdog 3.                                           |
| Parameters           | <ul style="list-style-type: none"><li>• <b>ADCx:</b> ADC instance</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>               |

Reference Manual to  
LL API cross  
reference:

- ISR AWD3 LL\_ADC\_ClearFlag\_AWD3

### LL\_ADC\_IsActiveFlag\_MST\_ADRDY

|                      |                                                                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_MST_ADRDY(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                  |
| Function description | Get flag multimode ADC ready of the ADC master.                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"><li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>                                                                                                                        |

Reference Manual to  
LL API cross  
reference:

- CSR ADRDY\_MST LL\_ADC\_IsActiveFlag\_MST\_ADRDY

**LL\_ADC\_IsActiveFlag\_SLV\_ADRDY**

|                                             |                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_SLV_ADRDY (ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                   |
| Function description                        | Get flag multimode ADC ready of the ADC slave.                                                                                                                                                            |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to LL API cross reference: | CSR ADRDY_SLV LL_ADC_IsActiveFlag_SLV_ADRDY                                                                                                                                                               |

**LL\_ADC\_IsActiveFlag\_MST\_EOC**

|                                             |                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_MST_EOC (ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                     |
| Function description                        | Get flag multimode ADC group regular end of unitary conversion of the ADC master.                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to LL API cross reference: | CSR EOC_MST LL_ADC_IsActiveFlag_MST_EOC                                                                                                                                                                   |

**LL\_ADC\_IsActiveFlag\_SLV\_EOC**

|                                             |                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_SLV_EOC (ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                     |
| Function description                        | Get flag multimode ADC group regular end of unitary conversion of the ADC slave.                                                                                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to LL API cross reference: | CSR EOC_SLV LL_ADC_IsActiveFlag_SLV_EOC                                                                                                                                                                   |

**LL\_ADC\_IsActiveFlag\_MST\_EOS**

|                                                   |                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_MST_EOS(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                        |
| Function description                              | Get flag multimode ADC group regular end of sequence conversions of the ADC master.                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR EOS_MST LL_ADC_IsActiveFlag_MST_EOS</li> </ul>                                                                                                               |

**LL\_ADC\_IsActiveFlag\_SLV\_EOS**

|                                                   |                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_SLV_EOS(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                        |
| Function description                              | Get flag multimode ADC group regular end of sequence conversions of the ADC slave.                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR EOS_SLV LL_ADC_IsActiveFlag_SLV_EOS</li> </ul>                                                                                                               |

**LL\_ADC\_IsActiveFlag\_MST\_OVR**

|                                                   |                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_MST_OVR(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                        |
| Function description                              | Get flag multimode ADC group regular overrun of the ADC master.                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR OVR_MST LL_ADC_IsActiveFlag_MST_OVR</li> </ul>                                                                                                               |

**LL\_ADC\_IsActiveFlag\_SLV\_OVR**

|                                             |                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_SLV_OVR<br/>(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                 |
| Function description                        | Get flag multimode ADC group regular overrun of the ADC slave.                                                                                                                                            |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR OVR_SLV LL_ADC_IsActiveFlag_SLV_OVR</li> </ul>                                                                                                               |

**LL\_ADC\_IsActiveFlag\_MST\_EOSMP**

|                                             |                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_IsActiveFlag_MST_EOSMP (ADC_Common_TypeDef<br/>* ADCxy_COMMON)</code>                                                                                           |
| Function description                        | Get flag multimode ADC group regular end of sampling of the ADC master.                                                                                                                                   |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR EOSMP_MST LL_ADC_IsActiveFlag_MST_EOSMP</li> </ul>                                                                                                           |

**LL\_ADC\_IsActiveFlag\_SLV\_EOSMP**

|                                             |                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_IsActiveFlag_SLV_EOSMP (ADC_Common_TypeDef<br/>* ADCxy_COMMON)</code>                                                                                           |
| Function description                        | Get flag multimode ADC group regular end of sampling of the ADC slave.                                                                                                                                    |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR EOSMP_SLV LL_ADC_IsActiveFlag_SLV_EOSMP</li> </ul>                                                                                                           |

**LL\_ADC\_IsActiveFlag\_MST\_JEOC**

|                                             |                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_MST_JEOC(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                     |
| Function description                        | Get flag multimode ADC group injected end of unitary conversion of the ADC master.                                                                                                                        |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR JEOC_MST LL_ADC_IsActiveFlag_MST_JEOC</li> </ul>                                                                                                             |

**LL\_ADC\_IsActiveFlag\_SLV\_JEOC**

|                                             |                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_SLV_JEOC(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                     |
| Function description                        | Get flag multimode ADC group injected end of unitary conversion of the ADC slave.                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR JEOC_SLV LL_ADC_IsActiveFlag_SLV_JEOC</li> </ul>                                                                                                             |

**LL\_ADC\_IsActiveFlag\_MST\_JEOS**

|                                             |                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_MST_JEOS(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                     |
| Function description                        | Get flag multimode ADC group injected end of sequence conversions of the ADC master.                                                                                                                      |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR JEOS_MST LL_ADC_IsActiveFlag_MST_JEOS</li> </ul>                                                                                                             |

**LL\_ADC\_IsActiveFlag\_SLV\_JEOS**

|                      |                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_SLV_JEOS(ADC_Common_TypeDef * ADCxy_COMMON)</code> |
| Function description | Get flag multimode ADC group injected end of sequence conversions of the ADC slave.                   |

|                                                   |                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | CSR JEOS_SLV LL_ADC_IsActiveFlag_SLV_JEOS                                                                                                                                                                 |

### LL\_ADC\_IsActiveFlag\_MST\_JQOVF

|                                                   |                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_IsActiveFlag_MST_JQOVF (ADC_Common_TypeDef<br/>* ADCxy_COMMON)</code>                                                                                           |
| Function description                              | Get flag multimode ADC group injected context queue overflow of the ADC master.                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | CSR JQOVF_MST LL_ADC_IsActiveFlag_MST_JQOVF                                                                                                                                                               |

### LL\_ADC\_IsActiveFlag\_SLV\_JQOVF

|                                                   |                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_ADC_IsActiveFlag_SLV_JQOVF (ADC_Common_TypeDef<br/>* ADCxy_COMMON)</code>                                                                                           |
| Function description                              | Get flag multimode ADC group injected context queue overflow of the ADC slave.                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | CSR JQOVF_SLV LL_ADC_IsActiveFlag_SLV_JQOVF                                                                                                                                                               |

### LL\_ADC\_IsActiveFlag\_MST\_AWD1

|                      |                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_MST_AWD1<br/>(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                |
| Function description | Get flag multimode ADC analog watchdog 1 of the ADC master.                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |

- Reference Manual to LL API cross reference:
- CSR AWD1\_MST LL\_ADC\_IsActiveFlag\_MST\_AWD1

### **LL\_ADC\_IsActiveFlag\_SLV\_AWD1**

|                                             |                                                                                                                                                                                                        |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_SLV_AWD1(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                   |
| Function description                        | Get flag multimode analog watchdog 1 of the ADC slave.                                                                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>_LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                       |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CSR AWD1_SLV LL_ADC_IsActiveFlag_SLV_AWD1</li> </ul>                                                                                                            |

### **LL\_ADC\_IsActiveFlag\_MST\_AWD2**

|                                             |                                                                                                                                                                                                        |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_MST_AWD2(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                   |
| Function description                        | Get flag multimode ADC analog watchdog 2 of the ADC master.                                                                                                                                            |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>_LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                       |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CSR AWD2_MST LL_ADC_IsActiveFlag_MST_AWD2</li> </ul>                                                                                                            |

### **LL\_ADC\_IsActiveFlag\_SLV\_AWD2**

|                                             |                                                                                                                                                                                                        |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_SLV_AWD2(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                   |
| Function description                        | Get flag multimode ADC analog watchdog 2 of the ADC slave.                                                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>_LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                       |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CSR AWD2_SLV LL_ADC_IsActiveFlag_SLV_AWD2</li> </ul>                                                                                                            |

**LL\_ADC\_IsActiveFlag\_MST\_AWD3**

|                                                   |                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_MST_AWD3(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                     |
| Function description                              | Get flag multimode ADC analog watchdog 3 of the ADC master.                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR AWD3_MST LL_ADC_IsActiveFlag_MST_AWD3</li> </ul>                                                                                                             |

**LL\_ADC\_IsActiveFlag\_SLV\_AWD3**

|                                                   |                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_IsActiveFlag_SLV_AWD3(ADC_Common_TypeDef * ADCxy_COMMON)</code>                                                                                                     |
| Function description                              | Get flag multimode ADC analog watchdog 3 of the ADC slave.                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>__LL_ADC_COMMON_INSTANCE()</code>)</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR AWD3_SLV LL_ADC_IsActiveFlag_SLV_AWD3</li> </ul>                                                                                                             |

**LL\_ADC\_EnableIT\_ADRDY**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_EnableIT_ADRDY(ADC_TypeDef * ADCx)</code>           |
| Function description                              | Enable ADC ready.                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER ADRDYIE LL_ADC_EnableIT_ADRDY</li> </ul> |

**LL\_ADC\_EnableIT\_EOC**

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_EnableIT_EOC<br/>(ADC_TypeDef * ADCx)</code>    |
| Function description                              | Enable interruption ADC group regular end of unitary conversion.                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER EOCIE LL_ADC_EnableIT_EOC</li> </ul> |

**LL\_ADC\_EnableIT\_EOS**

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_EnableIT_EOS<br/>(ADC_TypeDef * ADCx)</code>    |
| Function description                              | Enable interruption ADC group regular end of sequence conversions.                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER EOSIE LL_ADC_EnableIT_EOS</li> </ul> |

**LL\_ADC\_EnableIT\_OVR**

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_EnableIT_OVR<br/>(ADC_TypeDef * ADCx)</code>    |
| Function description                              | Enable ADC group regular interruption overrun.                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER OVRIE LL_ADC_EnableIT_OVR</li> </ul> |

**LL\_ADC\_EnableIT\_EOSMP**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_EnableIT_EOSMP<br/>(ADC_TypeDef * ADCx)</code>      |
| Function description                              | Enable interruption ADC group regular end of sampling.                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER EOSMPIE LL_ADC_EnableIT_EOSMP</li> </ul> |

**LL\_ADC\_EnableIT\_JEOC**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_EnableIT_JEOC(ADC_TypeDef * ADCx)</code>          |
| Function description                              | Enable interruption ADC group injected end of unitary conversion.                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER JEOCIE LL_ADC_EnableIT_JEOC</li> </ul> |

**LL\_ADC\_EnableIT\_JEOS**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_EnableIT_JEOS(ADC_TypeDef * ADCx)</code>          |
| Function description                              | Enable interruption ADC group injected end of sequence conversions.                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER JEOSIE LL_ADC_EnableIT_JEOS</li> </ul> |

**LL\_ADC\_EnableIT\_JQOVF**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_EnableIT_JQOVF(ADC_TypeDef * ADCx)</code>           |
| Function description                              | Enable interruption ADC group injected context queue overflow.                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER JQOVFIE LL_ADC_EnableIT_JQOVF</li> </ul> |

**LL\_ADC\_EnableIT\_AWD1**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_EnableIT_AWD1(ADC_TypeDef * ADCx)</code>          |
| Function description                              | Enable interruption ADC analog watchdog 1.                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER AWD1IE LL_ADC_EnableIT_AWD1</li> </ul> |

**LL\_ADC\_EnableIT\_AWD2**

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <b><u>__STATIC_INLINE void LL_ADC_EnableIT_AWD2(ADC_TypeDef * ADCx)</u></b>       |
| Function description                              | Enable interruption ADC analog watchdog 2.                                        |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>ADCx:</b> ADC instance</li></ul>       |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• IER AWD2IE LL_ADC_EnableIT_AWD2</li></ul> |

**LL\_ADC\_EnableIT\_AWD3**

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <b><u>__STATIC_INLINE void LL_ADC_EnableIT_AWD3(ADC_TypeDef * ADCx)</u></b>       |
| Function description                              | Enable interruption ADC analog watchdog 3.                                        |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>ADCx:</b> ADC instance</li></ul>       |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• IER AWD3IE LL_ADC_EnableIT_AWD3</li></ul> |

**LL\_ADC\_DisableIT\_ADRDY**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <b><u>__STATIC_INLINE void LL_ADC_DisableIT_ADRDY(ADC_TypeDef * ADCx)</u></b>        |
| Function description                              | Disable interruption ADC ready.                                                      |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>ADCx:</b> ADC instance</li></ul>          |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• IER ADRDYIE LL_ADC_DisableIT_ADRDY</li></ul> |

**LL\_ADC\_DisableIT\_EOC**

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <b><u>__STATIC_INLINE void LL_ADC_DisableIT_EOC(ADC_TypeDef * ADCx)</u></b>      |
| Function description                              | Disable interruption ADC group regular end of unitary conversion.                |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>ADCx:</b> ADC instance</li></ul>      |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• IER EOCIE LL_ADC_DisableIT_EOC</li></ul> |

**LL\_ADC\_DisableIT\_EOS**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_DisableIT_EOS<br/>(ADC_TypeDef * ADCx)</code>    |
| Function description                              | Disable interruption ADC group regular end of sequence conversions.                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER EOSIE LL_ADC_DisableIT_EOS</li> </ul> |

**LL\_ADC\_DisableIT\_OVR**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_DisableIT_OVR<br/>(ADC_TypeDef * ADCx)</code>    |
| Function description                              | Disable interruption ADC group regular overrun.                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER OVRIE LL_ADC_DisableIT_OVR</li> </ul> |

**LL\_ADC\_DisableIT\_EOSMP**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_DisableIT_EOSMP<br/>(ADC_TypeDef * ADCx)</code>      |
| Function description                              | Disable interruption ADC group regular end of sampling.                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER EOSMPIE LL_ADC_DisableIT_EOSMP</li> </ul> |

**LL\_ADC\_DisableIT\_JEOC**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_ADC_DisableIT_JEOC<br/>(ADC_TypeDef * ADCx)</code>     |
| Function description                              | Disable interruption ADC group regular end of unitary conversion.                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER JEOCIE LL_ADC_DisableIT_JEOC</li> </ul> |

**LL\_ADC\_DisableIT\_JEOS**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <b><u>STATIC_INLINE void LL_ADC_DisableIT_JEOS(ADC_TypeDef * ADCx)</u></b>         |
| Function description                              | Disable interruption ADC group injected end of sequence conversions.               |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>ADCx:</b> ADC instance</li></ul>        |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• IER JEOSIE LL_ADC_DisableIT_JEOS</li></ul> |

**LL\_ADC\_DisableIT\_JQOVF**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <b><u>STATIC_INLINE void LL_ADC_DisableIT_JQOVF(ADC_TypeDef * ADCx)</u></b>          |
| Function description                              | Disable interruption ADC group injected context queue overflow.                      |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>ADCx:</b> ADC instance</li></ul>          |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• IER JQOVFIE LL_ADC_DisableIT_JQOVF</li></ul> |

**LL\_ADC\_DisableIT\_AWD1**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <b><u>STATIC_INLINE void LL_ADC_DisableIT_AWD1(ADC_TypeDef * ADCx)</u></b>         |
| Function description                              | Disable interruption ADC analog watchdog 1.                                        |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>ADCx:</b> ADC instance</li></ul>        |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• IER AWD1IE LL_ADC_DisableIT_AWD1</li></ul> |

**LL\_ADC\_DisableIT\_AWD2**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <b><u>STATIC_INLINE void LL_ADC_DisableIT_AWD2(ADC_TypeDef * ADCx)</u></b>         |
| Function description                              | Disable interruption ADC analog watchdog 2.                                        |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>ADCx:</b> ADC instance</li></ul>        |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• IER AWD2IE LL_ADC_DisableIT_AWD2</li></ul> |

**LL\_ADC\_DisableIT\_AWD3**

Function name **`_STATIC_INLINE void LL_ADC_DisableIT_AWD3(ADC_TypeDef * ADCx)`**

Function description Disable interruption ADC analog watchdog 3.

Parameters • **ADCx:** ADC instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
IER AWD3IE LL\_ADC\_DisableIT\_AWD3

**LL\_ADC\_IsEnabledIT\_ADRDY**

Function name **`_STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_ADRDY(ADC_TypeDef * ADCx)`**

Function description Get state of interruption ADC ready (0: interrupt disabled, 1: interrupt enabled).

Parameters • **ADCx:** ADC instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
IER ADRDYIE LL\_ADC\_IsEnabledIT\_ADRDY

**LL\_ADC\_IsEnabledIT\_EOC**

Function name **`_STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_EOC(ADC_TypeDef * ADCx)`**

Function description Get state of interruption ADC group regular end of unitary conversion (0: interrupt disabled, 1: interrupt enabled).

Parameters • **ADCx:** ADC instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
IER EOCIE LL\_ADC\_IsEnabledIT\_EOC

**LL\_ADC\_IsEnabledIT\_EOS**

Function name **`_STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_EOS(ADC_TypeDef * ADCx)`**

Function description Get state of interruption ADC group regular end of sequence conversions (0: interrupt disabled, 1: interrupt enabled).

Parameters • **ADCx:** ADC instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
IER EOSIE LL\_ADC\_IsEnabledIT\_EOS

**LL\_ADC\_IsEnabledIT\_OVR**

|                                                   |                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_OVR(ADC_TypeDef * ADCx)</code>                   |
| Function description                              | Get state of interruption ADC group regular overrun (0: interrupt disabled, 1: interrupt enabled). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER OVRIE LL_ADC_IsEnabledIT_OVR</li> </ul>               |

**LL\_ADC\_IsEnabledIT\_EOSMP**

|                                                   |                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_EOSMP(ADC_TypeDef * ADCx)</code>                         |
| Function description                              | Get state of interruption ADC group regular end of sampling (0: interrupt disabled, 1: interrupt enabled). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER EOSMPIE LL_ADC_IsEnabledIT_EOSMP</li> </ul>                   |

**LL\_ADC\_IsEnabledIT\_JEOC**

|                                                   |                                                                                                                       |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_JEOC(ADC_TypeDef * ADCx)</code>                                     |
| Function description                              | Get state of interruption ADC group injected end of unitary conversion (0: interrupt disabled, 1: interrupt enabled). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER JEOCIE LL_ADC_IsEnabledIT_JEOC</li> </ul>                                |

**LL\_ADC\_IsEnabledIT\_JEOS**

|                                     |                                                                                                                         |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------|
| Function name                       | <code>__STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_JEOS(ADC_TypeDef * ADCx)</code>                                       |
| Function description                | Get state of interruption ADC group injected end of sequence conversions (0: interrupt disabled, 1: interrupt enabled). |
| Parameters                          | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                           |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                      |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• IER JEOSIE LL_ADC_IsEnabledIT_JEOS</li> </ul>                                  |

reference:

### **LL\_ADC\_IsEnabledIT\_JQOVF**

|                                                   |                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_JQOVF(ADC_TypeDef * ADCx)</code></b>                                          |
| Function description                              | Get state of interruption ADC group injected context queue overflow interrupt state (0: interrupt disabled, 1: interrupt enabled). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER JQOVFIE LL_ADC_IsEnabledIT_JQOVF</li> </ul>                                           |

### **LL\_ADC\_IsEnabledIT\_AWD1**

|                                                   |                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_AWD1(ADC_TypeDef * ADCx)</code></b>       |
| Function description                              | Get state of interruption ADC analog watchdog 1 (0: interrupt disabled, 1: interrupt enabled). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER AWD1IE LL_ADC_IsEnabledIT_AWD1</li> </ul>         |

### **LL\_ADC\_IsEnabledIT\_AWD2**

|                                                   |                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_AWD2(ADC_TypeDef * ADCx)</code></b>           |
| Function description                              | Get state of interruption Get ADC analog watchdog 2 (0: interrupt disabled, 1: interrupt enabled). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER AWD2IE LL_ADC_IsEnabledIT_AWD2</li> </ul>             |

### **LL\_ADC\_IsEnabledIT\_AWD3**

|                      |                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_ADC_IsEnabledIT_AWD3(ADC_TypeDef * ADCx)</code></b>           |
| Function description | Get state of interruption Get ADC analog watchdog 3 (0: interrupt disabled, 1: interrupt enabled). |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> </ul>                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                 |

Reference Manual to  
LL API cross  
reference:

- IER AWD3IE LL\_ADC\_IsEnabledIT\_AWD3

### **LL\_ADC\_CommonDeInit**

|                      |                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_ADC_CommonDeInit (ADC_Common_TypeDef * ADCxy_COMMON)</b>                                                                                                                                                                                                                                                                                                                   |
| Function description | De-initialize registers of all ADC instances belonging to the same ADC common instance to their default reset values.                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>_LL_ADC_COMMON_INSTANCE()</code>)</li> </ul>                                                                                                                                                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: ADC common registers are de-initialized</li> <li>– ERROR: not applicable</li> </ul> </li> </ul>                                                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>• This function is performing a hard reset, using high level clock source RCC ADC reset. Caution: On this STM32 serie, if several ADC instances are available on the selected device, RCC ADC reset will reset all ADC instances belonging to the common ADC instance. To de-initialize only 1 ADC instance, use function LL_ADC_DeInit().</li> </ul> |

### **LL\_ADC\_CommonInit**

|                      |                                                                                                                                                                                                                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_ADC_CommonInit (ADC_Common_TypeDef * ADCxy_COMMON, LL_ADC_CommonInitTypeDef * ADC_CommonInitStruct)</b>                                                                                                                                                                                    |
| Function description | Initialize some features of ADC common parameters (all ADC instances belonging to the same ADC common instance) and multimode (for devices with several ADC instances available).                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCxy_COMMON:</b> ADC common instance (can be set directly from CMSIS definition or by using helper macro <code>_LL_ADC_COMMON_INSTANCE()</code>)</li> <li>• <b>ADC_CommonInitStruct:</b> Pointer to a <code>LL_ADC_CommonInitStruct</code> structure</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: ADC common registers are initialized</li> <li>– ERROR: ADC common registers are not initialized</li> </ul> </li> </ul>                                        |
| Notes                | <ul style="list-style-type: none"> <li>• The setting of ADC common parameters is conditioned to ADC instances state: All ADC instances belonging to the same ADC common instance must be disabled.</li> </ul>                                                                                                |

### **LL\_ADC\_CommonStructInit**

|                      |                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_ADC_CommonStructInit (LL_ADC_CommonInitTypeDef * ADC_CommonInitStruct)</b>         |
| Function description | Set each LL_ADC_CommonInitStruct field to default value.                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADC_CommonInitStruct:</b> Pointer to a</li> </ul> |

LL\_ADC\_CommonInitTypeDef structure whose fields will be set to default values.

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>LL_ADC_Delnit</b> |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function name        | <b>ErrorStatus LL_ADC_Delnit (ADC_TypeDef * ADCx)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | De-initialize registers of the selected ADC instance to their default reset values.                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Return values        | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>SUCCESS: ADC registers are de-initialized</li> <li>ERROR: ADC registers are not de-initialized</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>To reset all ADC instances quickly (perform a hard reset), use function LL_ADC_CommonDelnit().</li> <li>If this functions returns error status, it means that ADC instance is in an unknown state. In this case, perform a hard reset using high level clock source RCC ADC reset. Caution: On this STM32 serie, if several ADC instances are available on the selected device, RCC ADC reset will reset all ADC instances belonging to the common ADC instance. Refer to function LL_ADC_CommonDelnit().</li> </ul> |

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_ADC_Init (ADC_TypeDef * ADCx,<br/>LL_ADC_InitTypeDef * ADC_InitStruct)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Initialize some features of ADC instance.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li><b>ADCx:</b> ADC instance</li> <li><b>ADC_InitStruct:</b> Pointer to a LL_ADC_REG_InitTypeDef structure</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>SUCCESS: ADC registers are initialized</li> <li>ERROR: ADC registers are not initialized</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>These parameters have an impact on ADC scope: ADC instance. Affects both group regular and group injected (availability of ADC group injected depends on STM32 families). Refer to corresponding unitary functions into Configuration of ADC hierarchical scope: ADC instance .</li> <li>The setting of these parameters by function LL_ADC_Init() is conditioned to ADC state: ADC instance must be disabled. This condition is applied to all ADC features, for efficiency and compatibility over all STM32 families. However, the different features can be set under different ADC state conditions (setting possible with ADC enabled without conversion on going, ADC enabled with conversion on going, ...) Each feature can be updated afterwards with a unitary function and potentially with ADC in a different state than disabled, refer to description of each function for setting conditioned to ADC state.</li> </ul> |

- After using this function, some other features must be configured using LL unitary functions. The minimum configuration remaining to be done is: Set ADC group regular or group injected sequencer: map channel on the selected sequencer rank. Refer to function `LL_ADC_REG_SetSequencerRanks()`. Set ADC channel sampling time Refer to function `LL_ADC_SetChannelSamplingTime();`

### **LL\_ADC\_StructInit**

|                      |                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void LL_ADC_StructInit (LL_ADC_InitTypeDef * ADC_InitStruct)</code>                                                                                        |
| Function description | Set each LL_ADC_InitTypeDef field to default value.                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADC_InitStruct:</b> Pointer to a LL_ADC_InitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                  |

### **LL\_ADC\_REG\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_ADC_REG_Init (ADC_TypeDef * ADCx, LL_ADC_REG_InitTypeDef * ADC_REG_InitStruct)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Initialize some features of ADC group regular.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>ADC_REG_InitStruct:</b> Pointer to a LL_ADC_REG_InitTypeDef structure</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: ADC registers are initialized</li> <li>– ERROR: ADC registers are not initialized</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Notes                | <ul style="list-style-type: none"> <li>• These parameters have an impact on ADC scope: ADC group regular. Refer to corresponding unitary functions into Configuration of ADC hierarchical scope: group regular (functions with prefix "REG").</li> <li>• The setting of these parameters by function <code>LL_ADC_Init()</code> is conditioned to ADC state: ADC instance must be disabled. This condition is applied to all ADC features, for efficiency and compatibility over all STM32 families. However, the different features can be set under different ADC state conditions (setting possible with ADC enabled without conversion on going, ADC enabled with conversion on going, ...) Each feature can be updated afterwards with a unitary function and potentially with ADC in a different state than disabled, refer to description of each function for setting conditioned to ADC state.</li> <li>• After using this function, other features must be configured using LL unitary functions. The minimum configuration remaining to be done is: Set ADC group regular or group injected sequencer: map channel on the selected sequencer rank. Refer to function <code>LL_ADC_REG_SetSequencerRanks()</code>. Set ADC channel sampling time Refer to function <code>LL_ADC_SetChannelSamplingTime();</code></li> </ul> |

---

```
LL_ADC_SetChannelSamplingTime();
```

### **LL\_ADC\_REG\_StructInit**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_ADC_REG_StructInit (LL_ADC_REG_InitTypeDef * ADC_REG_InitStruct)</b>                                                                                          |
| Function description | Set each LL_ADC_REG_InitTypeDef field to default value.                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADC_REG_InitStruct:</b> Pointer to a LL_ADC_REG_InitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                          |

### **LL\_ADC\_INJ\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_ADC_INJ_Init (ADC_TypeDef * ADCx, LL_ADC_INJ_InitTypeDef * ADC_INJ_InitStruct)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Initialize some features of ADC group injected.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> ADC instance</li> <li>• <b>ADC_INJ_InitStruct:</b> Pointer to a LL_ADC_INJ_InitTypeDef structure</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: ADC registers are initialized</li> <li>– ERROR: ADC registers are not initialized</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Notes                | <ul style="list-style-type: none"> <li>• These parameters have an impact on ADC scope: ADC group injected. Refer to corresponding unitary functions into Configuration of ADC hierarchical scope: group regular (functions with prefix "INJ").</li> <li>• The setting of these parameters by function LL_ADC_Init() is conditioned to ADC state: ADC instance must be disabled. This condition is applied to all ADC features, for efficiency and compatibility over all STM32 families. However, the different features can be set under different ADC state conditions (setting possible with ADC enabled without conversion on going, ADC enabled with conversion on going, ...) Each feature can be updated afterwards with a unitary function and potentially with ADC in a different state than disabled, refer to description of each function for setting conditioned to ADC state.</li> <li>• After using this function, other features must be configured using LL unitary functions. The minimum configuration remaining to be done is: Set ADC group injected sequencer: map channel on the selected sequencer rank. Refer to function LL_ADC_INJ_SetSequencerRanks(). Set ADC channel sampling time Refer to function LL_ADC_SetChannelSamplingTime();</li> <li>• Caution to ADC group injected contexts queue: On this STM32 serie, using successively several times this function will appear has having no effect. This is due to ADC group injected contexts queue (this feature cannot be disabled on this STM32 serie). To set several features of ADC group</li> </ul> |

injected, use function LL\_ADC\_INJ\_ConfigQueueContext().

### **LL\_ADC\_INJ\_StructInit**

|                      |                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_ADC_INJ_StructInit (LL_ADC_INJ_InitTypeDef *<br/>ADC_INJ_InitStruct)</b>                                                                                      |
| Function description | Set each LL_ADC_INJ_InitTypeDef field to default value.                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADC_INJ_InitStruct:</b> Pointer to a LL_ADC_INJ_InitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                          |

## 59.3 ADC Firmware driver defines

### 59.3.1 ADC

#### *Analog watchdog - Monitored channels*

|                                 |                                                                                                                                                |
|---------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_AWD_DISABLE              | ADC analog watchdog monitoring disabled                                                                                                        |
| LL_ADC_AWD_ALL_CHANNELS_REG     | ADC analog watchdog monitoring of all channels, converted by group regular only                                                                |
| LL_ADC_AWD_ALL_CHANNELS_INJ     | ADC analog watchdog monitoring of all channels, converted by group injected only                                                               |
| LL_ADC_AWD_ALL_CHANNELS_REG_INJ | ADC analog watchdog monitoring of all channels, converted by either group regular or injected                                                  |
| LL_ADC_AWD_CHANNEL_0_REG        | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN0, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_0_INJ        | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN0, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_0_REG_INJ    | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN0, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_1_REG        | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN1, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_1_INJ        | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN1, converted by group injected only              |

|                              |                                                                                                                                                |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_AWD_CHANNEL_1_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN1, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_2_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN2, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_2_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN2, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_2_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN2, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_3_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN3, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_3_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN3, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_3_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN3, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_4_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN4, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_4_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN4, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_4_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN4, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_5_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN5, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_5_INJ     | ADC analog watchdog monitoring of ADC external channel (channel                                                                                |

|                              |                                                                                                                                                |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
|                              | connected to GPIO pin) ADCx_IN5, converted by group injected only                                                                              |
| LL_ADC_AWD_CHANNEL_5_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN5, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_6_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN6, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_6_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN6, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_6_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN6, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_7_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN7, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_7_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN7, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_7_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN7, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_8_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN8, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_8_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN8, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_8_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN8, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_9_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN9,                                               |

|                               |                                                                                                                                                                      |
|-------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_AWD_CHANNEL_9_INJ      | converted by group regular only<br>ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN9, converted by group injected only |
| LL_ADC_AWD_CHANNEL_9_REG_INJ  | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN9, converted by either group regular or injected                       |
| LL_ADC_AWD_CHANNEL_10_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN10, converted by group regular only                                    |
| LL_ADC_AWD_CHANNEL_10_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN10, converted by group injected only                                   |
| LL_ADC_AWD_CHANNEL_10_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN10, converted by either group regular or injected                      |
| LL_ADC_AWD_CHANNEL_11_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN11, converted by group regular only                                    |
| LL_ADC_AWD_CHANNEL_11_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN11, converted by group injected only                                   |
| LL_ADC_AWD_CHANNEL_11_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN11, converted by either group regular or injected                      |
| LL_ADC_AWD_CHANNEL_12_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN12, converted by group regular only                                    |
| LL_ADC_AWD_CHANNEL_12_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN12, converted by group injected only                                   |
| LL_ADC_AWD_CHANNEL_12_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN12, converted by either group regular or injected                      |

|                               |                                                                                                                                                 |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_AWD_CHANNEL_13_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN13, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_13_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN13, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_13_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN13, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_14_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN14, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_14_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN14, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_14_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN14, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_15_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN15, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_15_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN15, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_15_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN15, converted by either group regular or injected |
| LL_ADC_AWD_CHANNEL_16_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN16, converted by group regular only               |
| LL_ADC_AWD_CHANNEL_16_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN16, converted by group injected only              |
| LL_ADC_AWD_CHANNEL_16_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN16, converted by either group regular or injected |

|                               |                                                                                                                                                        |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
|                               | converted by either group regular or injected                                                                                                          |
| LL_ADC_AWD_CHANNEL_17_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN17, converted by group regular only                      |
| LL_ADC_AWD_CHANNEL_17_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN17, converted by group injected only                     |
| LL_ADC_AWD_CHANNEL_17_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN17, converted by either group regular or injected        |
| LL_ADC_AWD_CHANNEL_18_REG     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN18, converted by group regular only                      |
| LL_ADC_AWD_CHANNEL_18_INJ     | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN18, converted by group injected only                     |
| LL_ADC_AWD_CHANNEL_18_REG_INJ | ADC analog watchdog monitoring of ADC external channel (channel connected to GPIO pin) ADCx_IN18, converted by either group regular or injected        |
| LL_ADC_AWD_CH_VREFINT_REG     | ADC analog watchdog monitoring of ADC internal channel connected to VrefInt: Internal voltage reference, converted by group regular only               |
| LL_ADC_AWD_CH_VREFINT_INJ     | ADC analog watchdog monitoring of ADC internal channel connected to VrefInt: Internal voltage reference, converted by group injected only              |
| LL_ADC_AWD_CH_VREFINT_REG_INJ | ADC analog watchdog monitoring of ADC internal channel connected to VrefInt: Internal voltage reference, converted by either group regular or injected |
| LL_ADC_AWD_CH_TEMPSENSOR_REG  | ADC analog watchdog monitoring of ADC internal channel connected to Temperature sensor, converted by group regular only                                |
| LL_ADC_AWD_CH_TEMPSENSOR_INJ  | ADC analog watchdog monitoring of ADC internal channel connected to Temperature sensor, converted by group injected only                               |

|                                  |                                                                                                                                                                                                  |
|----------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_AWD_CH_TEMPSENSOR_REG_INJ | ADC analog watchdog monitoring of ADC internal channel connected to Temperature sensor, converted by either group regular or injected                                                            |
| LL_ADC_AWD_CH_VBAT_REG           | ADC analog watchdog monitoring of ADC internal channel connected to Vbat/3: Vbat voltage through a divider ladder of factor 1/3 to have Vbat always below Vdda, converted by group regular only  |
| LL_ADC_AWD_CH_VBAT_INJ           | ADC analog watchdog monitoring of ADC internal channel connected to Vbat/3: Vbat voltage through a divider ladder of factor 1/3 to have Vbat always below Vdda, converted by group injected only |
| LL_ADC_AWD_CH_VBAT_REG_INJ       | ADC analog watchdog monitoring of ADC internal channel connected to Vbat/3: Vbat voltage through a divider ladder of factor 1/3 to have Vbat always below Vdda                                   |
| LL_ADC_AWD_CH_VOPAMP1_REG        | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC2, converted by group regular only                                                    |
| LL_ADC_AWD_CH_VOPAMP1_INJ        | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC2, converted by group injected only                                                   |
| LL_ADC_AWD_CH_VOPAMP1_REG_INJ    | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC2, converted by either group regular or injected                                      |
| LL_ADC_AWD_CH_VOPAMP2_REG        | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC2, converted by group regular only                                                    |
| LL_ADC_AWD_CH_VOPAMP2_INJ        | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC2, converted by group injected only                                                   |
| LL_ADC_AWD_CH_VOPAMP2_REG_INJ    | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC2, converted by either group regular or injected                                      |
| LL_ADC_AWD_CH_VOPAMP3_REG        | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to                                                                                          |

|                               |                                                                                                                                                             |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                               | ADC3, converted by group regular only                                                                                                                       |
| LL_ADC_AWD_CH_VOPAMP3_INJ     | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC3, converted by group injected only              |
| LL_ADC_AWD_CH_VOPAMP3_REG_INJ | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC3, converted by either group regular or injected |
| LL_ADC_AWD_CH_VOPAMP4_REG     | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC3, converted by group regular only               |
| LL_ADC_AWD_CH_VOPAMP4_INJ     | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC3, converted by group injected only              |
| LL_ADC_AWD_CH_VOPAMP4_REG_INJ | ADC analog watchdog monitoring of ADC internal channel connected to DAC1 channel 1, channel specific to ADC3, converted by either group regular or injected |

#### **Analog watchdog - Analog watchdog number**

|             |                              |
|-------------|------------------------------|
| LL_ADC_AWD1 | ADC analog watchdog number 1 |
| LL_ADC_AWD2 | ADC analog watchdog number 2 |
| LL_ADC_AWD3 | ADC analog watchdog number 3 |

#### **Analog watchdog - Thresholds**

|                                |                                                                                  |
|--------------------------------|----------------------------------------------------------------------------------|
| LL_ADC_AWD_THRESHOLD_HIGH      | ADC analog watchdog threshold high                                               |
| LL_ADC_AWD_THRESHOLD_LOW       | ADC analog watchdog threshold low                                                |
| LL_ADC_AWD_THRESHOLDS_HIGH_LOW | ADC analog watchdog both thresholds high and low concatenated into the same data |

#### **ADC instance - Channel number**

|                  |                                                               |
|------------------|---------------------------------------------------------------|
| LL_ADC_CHANNEL_0 | ADC external channel (channel connected to GPIO pin) ADCx_IN0 |
| LL_ADC_CHANNEL_1 | ADC external channel (channel connected to GPIO pin) ADCx_IN1 |
| LL_ADC_CHANNEL_2 | ADC external channel (channel connected to GPIO pin) ADCx_IN2 |
| LL_ADC_CHANNEL_3 | ADC external channel (channel connected to GPIO pin) ADCx_IN3 |
| LL_ADC_CHANNEL_4 | ADC external channel (channel connected to GPIO pin) ADCx_IN4 |
| LL_ADC_CHANNEL_5 | ADC external channel (channel connected to GPIO pin) ADCx_IN5 |

|                           |                                                                                                                                                                                                                       |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_CHANNEL_6          | ADC external channel (channel connected to GPIO pin) ADCx_IN6                                                                                                                                                         |
| LL_ADC_CHANNEL_7          | ADC external channel (channel connected to GPIO pin) ADCx_IN7                                                                                                                                                         |
| LL_ADC_CHANNEL_8          | ADC external channel (channel connected to GPIO pin) ADCx_IN8                                                                                                                                                         |
| LL_ADC_CHANNEL_9          | ADC external channel (channel connected to GPIO pin) ADCx_IN9                                                                                                                                                         |
| LL_ADC_CHANNEL_10         | ADC external channel (channel connected to GPIO pin) ADCx_IN10                                                                                                                                                        |
| LL_ADC_CHANNEL_11         | ADC external channel (channel connected to GPIO pin) ADCx_IN11                                                                                                                                                        |
| LL_ADC_CHANNEL_12         | ADC external channel (channel connected to GPIO pin) ADCx_IN12                                                                                                                                                        |
| LL_ADC_CHANNEL_13         | ADC external channel (channel connected to GPIO pin) ADCx_IN13                                                                                                                                                        |
| LL_ADC_CHANNEL_14         | ADC external channel (channel connected to GPIO pin) ADCx_IN14                                                                                                                                                        |
| LL_ADC_CHANNEL_15         | ADC external channel (channel connected to GPIO pin) ADCx_IN15                                                                                                                                                        |
| LL_ADC_CHANNEL_16         | ADC external channel (channel connected to GPIO pin) ADCx_IN16                                                                                                                                                        |
| LL_ADC_CHANNEL_17         | ADC external channel (channel connected to GPIO pin) ADCx_IN17                                                                                                                                                        |
| LL_ADC_CHANNEL_18         | ADC external channel (channel connected to GPIO pin) ADCx_IN18                                                                                                                                                        |
| LL_ADC_CHANNEL_VREFINT    | ADC internal channel connected to VrefInt: Internal voltage reference. On STM32F3, ADC channel available only on all ADC instances, but only one ADC instance is allowed to be connected to VrefInt at the same time. |
| LL_ADC_CHANNEL_TEMPSENSOR | ADC internal channel connected to Temperature sensor. On STM32F3, ADC channel available only on ADC instance: ADC1.                                                                                                   |
| LL_ADC_CHANNEL_VBAT       | ADC internal channel connected to Vbat/3: Vbat voltage through a divider ladder of factor 1/3 to have Vbat always below Vdda. On STM32F3, ADC channel available only on ADC instance: ADC1.                           |
| LL_ADC_CHANNEL_VOPAMP1    | ADC internal channel connected to OPAMP1 output. On STM32F3, ADC channel available only on ADC instance: ADC1.                                                                                                        |
| LL_ADC_CHANNEL_VOPAMP2    | ADC internal channel connected to OPAMP2 output. On STM32F3, ADC channel available only on ADC instance: ADC2.                                                                                                        |
| LL_ADC_CHANNEL_VOPAMP3    | ADC internal channel connected to OPAMP3 output. On STM32F3, ADC channel available only                                                                                                                               |

on ADC instance: ADC3.

`LL_ADC_CHANNEL_VOPAMP4` ADC internal channel connected to OPAMP4 output. On STM32F3, ADC channel available only on ADC instance: ADC4.

#### ***Channel - Sampling time***

|                                              |                                      |
|----------------------------------------------|--------------------------------------|
| <code>LL_ADC_SAMPLINGTIME_1CYCLE_5</code>    | Sampling time 1.5 ADC clock cycle    |
| <code>LL_ADC_SAMPLINGTIME_2CYCLES_5</code>   | Sampling time 2.5 ADC clock cycles   |
| <code>LL_ADC_SAMPLINGTIME_4CYCLES_5</code>   | Sampling time 4.5 ADC clock cycles   |
| <code>LL_ADC_SAMPLINGTIME_7CYCLES_5</code>   | Sampling time 7.5 ADC clock cycles   |
| <code>LL_ADC_SAMPLINGTIME_19CYCLES_5</code>  | Sampling time 19.5 ADC clock cycles  |
| <code>LL_ADC_SAMPLINGTIME_61CYCLES_5</code>  | Sampling time 61.5 ADC clock cycles  |
| <code>LL_ADC_SAMPLINGTIME_181CYCLES_5</code> | Sampling time 181.5 ADC clock cycles |
| <code>LL_ADC_SAMPLINGTIME_601CYCLES_5</code> | Sampling time 601.5 ADC clock cycles |

#### ***Channel - Single or differential ending***

|                                            |                                                                                                             |
|--------------------------------------------|-------------------------------------------------------------------------------------------------------------|
| <code>LL_ADC_SINGLE_ENDED</code>           | ADC channel ending set to single ended (literal also used to set calibration mode)                          |
| <code>LL_ADC_DIFFERENTIAL_ENDED</code>     | ADC channel ending set to differential (literal also used to set calibration mode)                          |
| <code>LL_ADC_BOTH_SINGLE_DIFF_ENDED</code> | ADC channel ending set to both single ended and differential (literal used only to set calibration factors) |

#### ***ADC common - Clock source***

|                                          |                                                                           |
|------------------------------------------|---------------------------------------------------------------------------|
| <code>LL_ADC_CLOCK_SYNC_PCLK_DIV1</code> | ADC synchronous clock derived from AHB clock without prescaler            |
| <code>LL_ADC_CLOCK_SYNC_PCLK_DIV2</code> | ADC synchronous clock derived from AHB clock with prescaler division by 2 |
| <code>LL_ADC_CLOCK_SYNC_PCLK_DIV4</code> | ADC synchronous clock derived from AHB clock with prescaler division by 4 |
| <code>LL_ADC_CLOCK_ASYNC_DIV1</code>     | ADC asynchronous clock without prescaler                                  |

#### ***ADC common - Measurement path to internal channels***

|                                              |                                                             |
|----------------------------------------------|-------------------------------------------------------------|
| <code>LL_ADC_PATH_INTERNAL_NONE</code>       | ADC measurement pathes all disabled                         |
| <code>LL_ADC_PATH_INTERNAL_VREFINT</code>    | ADC measurement path to internal channel VrefInt            |
| <code>LL_ADC_PATH_INTERNAL_TEMPSENSOR</code> | ADC measurement path to internal channel temperature sensor |
| <code>LL_ADC_PATH_INTERNAL_VBAT</code>       | ADC measurement path to internal channel Vbat               |

#### ***ADC instance - Data alignment***

|                                      |                                                                                     |
|--------------------------------------|-------------------------------------------------------------------------------------|
| <code>LL_ADC_DATA_ALIGN_RIGHT</code> | ADC conversion data alignment: right aligned (alignment on data register LSB bit 0) |
| <code>LL_ADC_DATA_ALIGN_LEFT</code>  | ADC conversion data alignment: left aligned (aligment                               |

on data register MSB bit 15)

***ADC flags***

|                       |                                                                          |
|-----------------------|--------------------------------------------------------------------------|
| LL_ADC_FLAG_ADRDY     | ADC flag ADC instance ready                                              |
| LL_ADC_FLAG_EOC       | ADC flag ADC group regular end of unitary conversion                     |
| LL_ADC_FLAG_EOS       | ADC flag ADC group regular end of sequence conversions                   |
| LL_ADC_FLAG_OVR       | ADC flag ADC group regular overrun                                       |
| LL_ADC_FLAG_EOSMP     | ADC flag ADC group regular end of sampling phase                         |
| LL_ADC_FLAG_JEOC      | ADC flag ADC group injected end of unitary conversion                    |
| LL_ADC_FLAG_JEOS      | ADC flag ADC group injected end of sequence conversions                  |
| LL_ADC_FLAG_JQOVF     | ADC flag ADC group injected contexts queue overflow                      |
| LL_ADC_FLAG_AWD1      | ADC flag ADC analog watchdog 1                                           |
| LL_ADC_FLAG_AWD2      | ADC flag ADC analog watchdog 2                                           |
| LL_ADC_FLAG_AWD3      | ADC flag ADC analog watchdog 3                                           |
| LL_ADC_FLAG_ADRDY_MST | ADC flag ADC multimode master instance ready                             |
| LL_ADC_FLAG_ADRDY_SLV | ADC flag ADC multimode slave instance ready                              |
| LL_ADC_FLAG_EOC_MST   | ADC flag ADC multimode master group regular end of unitary conversion    |
| LL_ADC_FLAG_EOC_SLV   | ADC flag ADC multimode slave group regular end of unitary conversion     |
| LL_ADC_FLAG_EOS_MST   | ADC flag ADC multimode master group regular end of sequence conversions  |
| LL_ADC_FLAG_EOS_SLV   | ADC flag ADC multimode slave group regular end of sequence conversions   |
| LL_ADC_FLAG_OVR_MST   | ADC flag ADC multimode master group regular overrun                      |
| LL_ADC_FLAG_OVR_SLV   | ADC flag ADC multimode slave group regular overrun                       |
| LL_ADC_FLAG_EOSMP_MST | ADC flag ADC multimode master group regular end of sampling phase        |
| LL_ADC_FLAG_EOSMP_SLV | ADC flag ADC multimode slave group regular end of sampling phase         |
| LL_ADC_FLAG_JEOC_MST  | ADC flag ADC multimode master group injected end of unitary conversion   |
| LL_ADC_FLAG_JEOC_SLV  | ADC flag ADC multimode slave group injected end of unitary conversion    |
| LL_ADC_FLAG_JEOS_MST  | ADC flag ADC multimode master group injected end of sequence conversions |
| LL_ADC_FLAG_JEOS_SLV  | ADC flag ADC multimode slave group injected end of sequence conversions  |
| LL_ADC_FLAG_JQOVF_MST | ADC flag ADC multimode master group injected contexts queue overflow     |

|                       |                                                                     |
|-----------------------|---------------------------------------------------------------------|
| LL_ADC_FLAG_JQOVF_SLV | ADC flag ADC multimode slave group injected contexts queue overflow |
| LL_ADC_FLAG_AWD1_MST  | ADC flag ADC multimode master analog watchdog 1 of the ADC master   |
| LL_ADC_FLAG_AWD1_SLV  | ADC flag ADC multimode slave analog watchdog 1 of the ADC slave     |
| LL_ADC_FLAG_AWD2_MST  | ADC flag ADC multimode master analog watchdog 2 of the ADC master   |
| LL_ADC_FLAG_AWD2_SLV  | ADC flag ADC multimode slave analog watchdog 2 of the ADC slave     |
| LL_ADC_FLAG_AWD3_MST  | ADC flag ADC multimode master analog watchdog 3 of the ADC master   |
| LL_ADC_FLAG_AWD3_SLV  | ADC flag ADC multimode slave analog watchdog 3 of the ADC slave     |

***ADC instance - Groups***

|                               |                                                         |
|-------------------------------|---------------------------------------------------------|
| LL_ADC_GROUP_REGULAR          | ADC group regular (available on all STM32 devices)      |
| LL_ADC_GROUP_INJECTED         | ADC group injected (not available on all STM32 devices) |
| LL_ADC_GROUP_REGULAR_INJECTED | ADC both groups regular and injected                    |

***Definitions of ADC hardware constraints delays***

|                                      |                                                                        |
|--------------------------------------|------------------------------------------------------------------------|
| LL_ADC_DELAY_INTERNAL_REGUL_STAB_US  | Delay for ADC stabilization time (ADC voltage regulator start-up time) |
| LL_ADC_DELAY_VREFINT_STAB_US         | Delay for internal voltage reference stabilization time                |
| LL_ADC_DELAY_TEMPSENSOR_STAB_US      | Delay for temperature sensor stabilization time                        |
| LL_ADC_DELAY_CALIB_ENABLE_ADC_CYCLES | Delay required between ADC end of calibration and ADC enable           |

***ADC group injected - Context queue mode***

|                                        |  |
|----------------------------------------|--|
| LL_ADC_INJ_QUEUE_2CONTEXTS_LAST_ACTIVE |  |
| LL_ADC_INJ_QUEUE_2CONTEXTS_END_EMPTY   |  |

***ADC group injected - Sequencer discontinuous mode***

|                                |                                                                                              |
|--------------------------------|----------------------------------------------------------------------------------------------|
| LL_ADC_INJ_SEQ_DISCONT_DISABLE | ADC group injected sequencer discontinuous mode disable                                      |
| LL_ADC_INJ_SEQ_DISCONT_1RANK   | ADC group injected sequencer discontinuous mode enable with sequence interruption every rank |

***ADC group injected - Sequencer ranks***

|                   |                                     |
|-------------------|-------------------------------------|
| LL_ADC_INJ_RANK_1 | ADC group injected sequencer rank 1 |
| LL_ADC_INJ_RANK_2 | ADC group injected sequencer rank 2 |
| LL_ADC_INJ_RANK_3 | ADC group injected sequencer rank 3 |

`LL_ADC_INJ_RANK_4` ADC group injected sequencer rank 4

***ADC group injected - Sequencer scan length***

|                                                |                                                                                                            |
|------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| <code>LL_ADC_INJ_SEQ_SCAN_DISABLE</code>       | ADC group injected sequencer disable (equivalent to sequencer of 1 rank: ADC conversion on only 1 channel) |
| <code>LL_ADC_INJ_SEQ_SCAN_ENABLE_2RANKS</code> | ADC group injected sequencer enable with 2 ranks in the sequence                                           |
| <code>LL_ADC_INJ_SEQ_SCAN_ENABLE_3RANKS</code> | ADC group injected sequencer enable with 3 ranks in the sequence                                           |
| <code>LL_ADC_INJ_SEQ_SCAN_ENABLE_4RANKS</code> | ADC group injected sequencer enable with 4 ranks in the sequence                                           |

***ADC group injected - Trigger edge***

|                                                |                                                                                     |
|------------------------------------------------|-------------------------------------------------------------------------------------|
| <code>LL_ADC_INJ_TRIG_EXT_RISING</code>        | ADC group injected conversion trigger polarity set to rising edge                   |
| <code>LL_ADC_INJ_TRIG_EXT_FALLING</code>       | ADC group injected conversion trigger polarity set to falling edge                  |
| <code>LL_ADC_INJ_TRIG_EXT_RISINGFALLING</code> | ADC group injected conversion trigger polarity set to both rising and falling edges |

***ADC group injected - Trigger source***

|                                                  |                                                                                                                                                                                     |
|--------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>LL_ADC_INJ_TRIG_SOFTWARE</code>            | ADC group injected conversion trigger internal: SW start.. Trigger edge set to rising edge (default setting).                                                                       |
| <code>LL_ADC_INJ_TRIG_EXT_TIM1_TRGO</code>       | ADC group injected conversion trigger from external IP: TIM1 TRG0. Trigger edge set to rising edge (default setting).                                                               |
| <code>LL_ADC_INJ_TRIG_EXT_TIM1_TRGO</code>       | ADC group injected conversion trigger from external IP: TIM1 TRG0. Trigger edge set to rising edge (default setting).                                                               |
| <code>LL_ADC_INJ_TRIG_EXT_TIM1_CH4</code>        | ADC group injected conversion trigger from external IP: TIM1 channel 4 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| <code>LL_ADC_INJ_TRIG_EXT_TIM1_CH4</code>        | ADC group injected conversion trigger from external IP: TIM1 channel 4 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| <code>LL_ADC_INJ_TRIG_EXT_TIM2_TRGO_ADC12</code> | ADC group injected conversion trigger from external IP: TIM2 TRG0. Trigger edge set to rising edge (default setting).                                                               |
| <code>LL_ADC_INJ_TRIG_EXT_TIM2_CH1_ADC12</code>  | ADC group injected conversion trigger from external IP: TIM2 channel 1 event (capture compare: input capture or                                                                     |

|                                       |                                                                                                                                                                                     |
|---------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_INJ_TRIG_EXT_TIM3_CH4_ADC12    | output capture). Trigger edge set to rising edge (default setting).                                                                                                                 |
| LL_ADC_INJ_TRIG_EXT_TIM4_TRGO_ADC12   | ADC group injected conversion trigger from external IP: TIM4 TRG0. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_INJ_TRIG_EXT EXTI_LINE15_ADC12 | ADC group injected conversion trigger from external IP: external interrupt line 15. Trigger edge set to rising edge (default setting).                                              |
| LL_ADC_INJ_TRIG_EXT_TIM8_CH4_ADC12    | ADC group injected conversion trigger from external IP: TIM8 channel 4 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_INJ_TRIG_EXT_TIM1_TRGO2        | ADC group injected conversion trigger from external IP: TIM1 TRG02. Trigger edge set to rising edge (default setting).                                                              |
| LL_ADC_INJ_TRIG_EXT_TIM1_TRGO2        | ADC group injected conversion trigger from external IP: TIM1 TRG02. Trigger edge set to rising edge (default setting).                                                              |
| LL_ADC_INJ_TRIG_EXT_TIM8_TRGO         | ADC group injected conversion trigger from external IP: TIM8 TRG0. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_INJ_TRIG_EXT_TIM8_TRGO         | ADC group injected conversion trigger from external IP: TIM8 TRG0. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_INJ_TRIG_EXT_TIM8_TRGO2        | ADC group injected conversion trigger from external IP: TIM8 TRG02. Trigger edge set to rising edge (default setting).                                                              |
| LL_ADC_INJ_TRIG_EXT_TIM8_TRGO2        | ADC group injected conversion trigger from external IP: TIM8 TRG02. Trigger edge set to rising edge (default setting).                                                              |
| LL_ADC_INJ_TRIG_EXT_TIM3_CH3_ADC12    | ADC group injected conversion trigger from external IP: TIM3 channel 3 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |

|                                     |                                                                                                                                                                                     |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_INJ_TRIG_EXT_TIM3_TRGO       | ADC group injected conversion trigger from external IP: TIM3 TRG0. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_INJ_TRIG_EXT_TIM3_TRGO       | ADC group injected conversion trigger from external IP: TIM3 TRG0. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_INJ_TRIG_EXT_TIM3_CH1_ADC12  | ADC group injected conversion trigger from external IP: TIM3 channel 1 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_INJ_TRIG_EXT_TIM6_TRGO_ADC12 | ADC group injected conversion trigger from external IP: TIM6 TRG0. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_INJ_TRIG_EXT_TIM15_TRGO      | ADC group injected conversion trigger from external IP: TIM15 TRG0. Trigger edge set to rising edge (default setting).                                                              |
| LL_ADC_INJ_TRIG_EXT_TIM15_TRGO      | ADC group injected conversion trigger from external IP: TIM15 TRG0. Trigger edge set to rising edge (default setting).                                                              |
| LL_ADC_INJ_TRIG_EXT_TIM4_CH3_ADC34  | ADC group injected conversion trigger from external IP: TIM4 channel 3 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_INJ_TRIG_EXT_TIM8_CH2_ADC34  | ADC group injected conversion trigger from external IP: TIM8 channel 2 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_INJ_TRIG_EXT_TIM8_CH4_ADC34  | ADC group injected conversion trigger from external IP: TIM8 channel 4 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_INJ_TRIG_EXT_TIM4_CH4_ADC34  | ADC group injected conversion trigger from external IP: TIM4 channel 4 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_INJ_TRIG_EXT_TIM4_TRGO_ADC34 | ADC group injected conversion trigger from external IP: TIM4 TRG0. Trigger edge set to rising edge (default setting).                                                               |

|                                     |                                                                                                                                                                                     |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_INJ_TRIG_EXT_TIM1_CH3_ADC34  | ADC group injected conversion trigger from external IP: TIM1 channel 3 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_INJ_TRIG_EXT_TIM2_TRGO_ADC34 | ADC group injected conversion trigger from external IP: TIM2 TRG0. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_INJ_TRIG_EXT_TIM7_TRGO_ADC34 | ADC group injected conversion trigger from external IP: TIM7 TRG0. Trigger edge set to rising edge (default setting).                                                               |

***ADC group injected - Automatic trigger mode***

|                                  |                                                                                                                                                                                                                                                                                                 |
|----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_INJ_TRIG_INDEPENDENT      | ADC group injected conversion trigger independent. Setting mandatory if ADC group injected injected trigger source is set to an external trigger.                                                                                                                                               |
| LL_ADC_INJ_TRIG_FROM_GRP_REGULAR | ADC group injected conversion trigger from ADC group regular. Setting compliant only with group injected trigger source set to SW start, without any further action on ADC group injected conversion start or stop: in this case, ADC group injected is controlled only from ADC group regular. |

***ADC interruptions for configuration (interruption enable or disable)***

|                 |                                                                 |
|-----------------|-----------------------------------------------------------------|
| LL_ADC_IT_ADRDY | ADC interruption ADC instance ready                             |
| LL_ADC_IT_EOC   | ADC interruption ADC group regular end of unitary conversion    |
| LL_ADC_IT_EOS   | ADC interruption ADC group regular end of sequence conversions  |
| LL_ADC_IT_OVR   | ADC interruption ADC group regular overrun                      |
| LL_ADC_IT_EOSMP | ADC interruption ADC group regular end of sampling phase        |
| LL_ADC_IT_JEOC  | ADC interruption ADC group injected end of unitary conversion   |
| LL_ADC_IT_JEOS  | ADC interruption ADC group injected end of sequence conversions |
| LL_ADC_IT_JQOVF | ADC interruption ADC group injected contexts queue overflow     |
| LL_ADC_IT_AWD1  | ADC interruption ADC analog watchdog 1                          |
| LL_ADC_IT_AWD2  | ADC interruption ADC analog watchdog 2                          |
| LL_ADC_IT_AWD3  | ADC interruption ADC analog watchdog 3                          |

***ADC instance - Low power mode***

|                     |                                                                                                                                                                                     |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_LP_MODE_NONE | No ADC low power mode activated                                                                                                                                                     |
| LL_ADC_LP_AUTOWAIT  | ADC low power mode auto delay: Dynamic low power mode, ADC conversions are performed only when necessary (when previous ADC conversion data is read). See description with function |

***Multimode - DMA transfer*****LL\_ADC\_MULTI\_REG\_DMA\_EACH\_ADC**

ADC multimode group regular conversions are transferred by DMA: each ADC uses its own DMA channel, with its individual DMA transfer settings

**LL\_ADC\_MULTI\_REG\_DMA\_LIMIT\_RES12\_10B**

ADC multimode group regular conversions are transferred by DMA, one DMA channel for both ADC (DMA of ADC master), in limited mode (one shot mode): DMA transfer requests are stopped when number of DMA data transfers (number of ADC conversions) is reached. This ADC mode is intended to be used with DMA mode non-circular. Setting for ADC resolution of 12 and 10 bits

**LL\_ADC\_MULTI\_REG\_DMA\_LIMIT\_RES8\_6B**

ADC multimode group regular conversions are transferred by DMA, one DMA channel for both ADC (DMA of ADC master), in limited mode (one shot mode): DMA transfer requests are stopped when number of DMA data transfers (number of ADC conversions) is reached. This ADC mode is intended to be used with DMA mode non-circular. Setting for ADC resolution of 8 and 6 bits

**LL\_ADC\_MULTI\_REG\_DMA\_UNLMT\_RES12\_10B**

ADC multimode group regular conversions are transferred by DMA, one DMA channel for both ADC (DMA of ADC master), in unlimited mode: DMA transfer requests are unlimited, whatever number of DMA data transferred (number of ADC conversions). This ADC mode is intended to be used with DMA mode circular. Setting for ADC resolution of 12 and 10 bits

**LL\_ADC\_MULTI\_REG\_DMA\_UNLMT\_RES8\_6B**

ADC multimode group regular conversions are transferred by DMA, one DMA channel for both ADC (DMA of ADC master), in unlimited mode: DMA transfer requests are unlimited, whatever number of DMA data transferred (number of ADC conversions). This ADC mode is intended to be used with DMA mode circular. Setting for ADC resolution of 8 and 6 bits

***Multimode - ADC master or slave***

|                           |                                                                                    |
|---------------------------|------------------------------------------------------------------------------------|
| LL_ADC_MULTI_MASTER       | In multimode, selection among several ADC instances: ADC master                    |
| LL_ADC_MULTI_SLAVE        | In multimode, selection among several ADC instances: ADC slave                     |
| LL_ADC_MULTI_MASTER_SLAVE | In multimode, selection among several ADC instances: both ADC master and ADC slave |

***Multimode - Mode***

|                                   |                                                                                                                    |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------------|
| LL_ADC_MULTI_INDEPENDENT          | ADC dual mode disabled (ADC independent mode)                                                                      |
| LL_ADC_MULTI_DUAL_REG_SIMULT      | ADC dual mode enabled: group regular simultaneous                                                                  |
| LL_ADC_MULTI_DUAL_REG_INTERL      | ADC dual mode enabled: Combined group regular interleaved                                                          |
| LL_ADC_MULTI_DUAL_INJ_SIMULT      | ADC dual mode enabled: group injected simultaneous                                                                 |
| LL_ADC_MULTI_DUAL_INJ_ALTERN      | ADC dual mode enabled: group injected alternate trigger. Works only with external triggers (not internal SW start) |
| LL_ADC_MULTI_DUAL_REG_SIM_INJ_SIM | ADC dual mode enabled: Combined group regular simultaneous + group injected simultaneous                           |
| LL_ADC_MULTI_DUAL_REG_SIM_INJ_ALT | ADC dual mode enabled: Combined group regular simultaneous + group injected alternate trigger                      |
| LL_ADC_MULTI_DUAL_REG_INT_INJ_SIM | ADC dual mode enabled: Combined group regular interleaved + group injected simultaneous                            |

***Multimode - Delay between two sampling phases***

|                                   |                                                                     |
|-----------------------------------|---------------------------------------------------------------------|
| LL_ADC_MULTI_TWOSMP_DELAY_1CYCLE  | ADC multimode delay between two sampling phases: 1 ADC clock cycle  |
| LL_ADC_MULTI_TWOSMP_DELAY_2CYCLES | ADC multimode delay between two sampling phases: 2 ADC clock cycles |
| LL_ADC_MULTI_TWOSMP_DELAY_3CYCLES | ADC multimode delay between two sampling phases: 3 ADC clock cycles |
| LL_ADC_MULTI_TWOSMP_DELAY_4CYCLES | ADC multimode delay between two sampling phases: 4 ADC clock cycles |
| LL_ADC_MULTI_TWOSMP_DELAY_5CYCLES | ADC multimode delay between two sampling phases: 5 ADC clock cycles |
| LL_ADC_MULTI_TWOSMP_DELAY_6CYCLES | ADC multimode delay between two sampling phases: 6 ADC clock cycles |
| LL_ADC_MULTI_TWOSMP_DELAY_7CYCLES | ADC multimode delay between two sampling phases: 7 ADC clock cycles |
| LL_ADC_MULTI_TWOSMP_DELAY_8CYCLES | ADC multimode delay between two                                     |

|                                    |                                                                                                            |
|------------------------------------|------------------------------------------------------------------------------------------------------------|
| LL_ADC_MULTI_TWOSMP_DELAY_9CYCLES  | sampling phases: 8 ADC clock cycles<br>ADC multimode delay between two sampling phases: 9 ADC clock cycles |
| LL_ADC_MULTI_TWOSMP_DELAY_10CYCLES | ADC multimode delay between two sampling phases: 10 ADC clock cycles                                       |
| LL_ADC_MULTI_TWOSMP_DELAY_11CYCLES | ADC multimode delay between two sampling phases: 11 ADC clock cycles                                       |
| LL_ADC_MULTI_TWOSMP_DELAY_12CYCLES | ADC multimode delay between two sampling phases: 12 ADC clock cycles                                       |

***ADC instance - Offset number***

|                 |                                                                                                                                                                           |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_OFFSET_1 | ADC offset number 1: ADC channel and offset level to which the offset programmed will be applied (independently of channel mapped on ADC group regular or group injected) |
| LL_ADC_OFFSET_2 | ADC offset number 2: ADC channel and offset level to which the offset programmed will be applied (independently of channel mapped on ADC group regular or group injected) |
| LL_ADC_OFFSET_3 | ADC offset number 3: ADC channel and offset level to which the offset programmed will be applied (independently of channel mapped on ADC group regular or group injected) |
| LL_ADC_OFFSET_4 | ADC offset number 4: ADC channel and offset level to which the offset programmed will be applied (independently of channel mapped on ADC group regular or group injected) |

***ADC instance - Offset state***

|                       |                                                                     |
|-----------------------|---------------------------------------------------------------------|
| LL_ADC_OFFSET_DISABLE | ADC offset disabled (among ADC selected offset number 1, 2, 3 or 4) |
| LL_ADC_OFFSET_ENABLE  | ADC offset enabled (among ADC selected offset number 1, 2, 3 or 4)  |

***ADC registers compliant with specific purpose***

|                                   |  |
|-----------------------------------|--|
| LL_ADC_DMA_REG_REGULAR_DATA       |  |
| LL_ADC_DMA_REG_REGULAR_DATA_MULTI |  |

***ADC group regular - Continuous mode***

|                            |                                                                                                                                      |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_REG_CONV_SINGLE     | ADC conversions are performed in single mode: one conversion per trigger                                                             |
| LL_ADC_REG_CONV_CONTINUOUS | ADC conversions are performed in continuous mode: after the first trigger, following conversions launched successively automatically |

***ADC group regular - DMA transfer of ADC conversion data***

|                                 |                                                                                                                                                                                                                       |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_REG_DMA_TRANSFER_NONE    | ADC conversions are not transferred by DMA                                                                                                                                                                            |
| LL_ADC_REG_DMA_TRANSFER_LIMITED | ADC conversion data are transferred by DMA, in limited mode (one shot mode): DMA transfer requests are stopped when number of DMA data transfers (number of ADC conversions) is reached. This ADC mode is intended to |

be used with DMA mode non-circular.

|                                   |                                                                                                                                                                                                                                       |
|-----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_ADC_REG_DMA_TRANSFER_UNLIMITED | ADC conversion data are transferred by DMA, in unlimited mode: DMA transfer requests are unlimited, whatever number of DMA data transferred (number of ADC conversions). This ADC mode is intended to be used with DMA mode circular. |
|-----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

#### ***ADC group regular - Overrun behavior on conversion data***

|                                 |                                                                 |
|---------------------------------|-----------------------------------------------------------------|
| LL_ADC_REG_OVR_DATA_PRESERVED   | ADC group regular behavior in case of overrun: data preserved   |
| LL_ADC_REG_OVR_DATA_OVERWRITTEN | ADC group regular behavior in case of overrun: data overwritten |

#### ***ADC group regular - Sequencer discontinuous mode***

|                                |                                                                                                 |
|--------------------------------|-------------------------------------------------------------------------------------------------|
| LL_ADC_REG_SEQ_DISCONT_DISABLE | ADC group regular sequencer discontinuous mode disable                                          |
| LL_ADC_REG_SEQ_DISCONT_1RANK   | ADC group regular sequencer discontinuous mode enable with sequence interruption every rank     |
| LL_ADC_REG_SEQ_DISCONT_2RANKS  | ADC group regular sequencer discontinuous mode enabled with sequence interruption every 2 ranks |
| LL_ADC_REG_SEQ_DISCONT_3RANKS  | ADC group regular sequencer discontinuous mode enable with sequence interruption every 3 ranks  |
| LL_ADC_REG_SEQ_DISCONT_4RANKS  | ADC group regular sequencer discontinuous mode enable with sequence interruption every 4 ranks  |
| LL_ADC_REG_SEQ_DISCONT_5RANKS  | ADC group regular sequencer discontinuous mode enable with sequence interruption every 5 ranks  |
| LL_ADC_REG_SEQ_DISCONT_6RANKS  | ADC group regular sequencer discontinuous mode enable with sequence interruption every 6 ranks  |
| LL_ADC_REG_SEQ_DISCONT_7RANKS  | ADC group regular sequencer discontinuous mode enable with sequence interruption every 7 ranks  |
| LL_ADC_REG_SEQ_DISCONT_8RANKS  | ADC group regular sequencer discontinuous mode enable with sequence interruption every 8 ranks  |

#### ***ADC group regular - Sequencer ranks***

|                   |                                    |
|-------------------|------------------------------------|
| LL_ADC_REG_RANK_1 | ADC group regular sequencer rank 1 |
| LL_ADC_REG_RANK_2 | ADC group regular sequencer rank 2 |
| LL_ADC_REG_RANK_3 | ADC group regular sequencer rank 3 |
| LL_ADC_REG_RANK_4 | ADC group regular sequencer rank 4 |

|                    |                                     |
|--------------------|-------------------------------------|
| LL_ADC_REG_RANK_5  | ADC group regular sequencer rank 5  |
| LL_ADC_REG_RANK_6  | ADC group regular sequencer rank 6  |
| LL_ADC_REG_RANK_7  | ADC group regular sequencer rank 7  |
| LL_ADC_REG_RANK_8  | ADC group regular sequencer rank 8  |
| LL_ADC_REG_RANK_9  | ADC group regular sequencer rank 9  |
| LL_ADC_REG_RANK_10 | ADC group regular sequencer rank 10 |
| LL_ADC_REG_RANK_11 | ADC group regular sequencer rank 11 |
| LL_ADC_REG_RANK_12 | ADC group regular sequencer rank 12 |
| LL_ADC_REG_RANK_13 | ADC group regular sequencer rank 13 |
| LL_ADC_REG_RANK_14 | ADC group regular sequencer rank 14 |
| LL_ADC_REG_RANK_15 | ADC group regular sequencer rank 15 |
| LL_ADC_REG_RANK_16 | ADC group regular sequencer rank 16 |

***ADC group regular - Sequencer scan length***

|                                    |                                                                                                                 |
|------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| LL_ADC_REG_SEQ_SCAN_DISABLE        | ADC group regular sequencer disable<br>(equivalent to sequencer of 1 rank:<br>ADC conversion on only 1 channel) |
| LL_ADC_REG_SEQ_SCAN_ENABLE_2RANKS  | ADC group regular sequencer enable<br>with 2 ranks in the sequence                                              |
| LL_ADC_REG_SEQ_SCAN_ENABLE_3RANKS  | ADC group regular sequencer enable<br>with 3 ranks in the sequence                                              |
| LL_ADC_REG_SEQ_SCAN_ENABLE_4RANKS  | ADC group regular sequencer enable<br>with 4 ranks in the sequence                                              |
| LL_ADC_REG_SEQ_SCAN_ENABLE_5RANKS  | ADC group regular sequencer enable<br>with 5 ranks in the sequence                                              |
| LL_ADC_REG_SEQ_SCAN_ENABLE_6RANKS  | ADC group regular sequencer enable<br>with 6 ranks in the sequence                                              |
| LL_ADC_REG_SEQ_SCAN_ENABLE_7RANKS  | ADC group regular sequencer enable<br>with 7 ranks in the sequence                                              |
| LL_ADC_REG_SEQ_SCAN_ENABLE_8RANKS  | ADC group regular sequencer enable<br>with 8 ranks in the sequence                                              |
| LL_ADC_REG_SEQ_SCAN_ENABLE_9RANKS  | ADC group regular sequencer enable<br>with 9 ranks in the sequence                                              |
| LL_ADC_REG_SEQ_SCAN_ENABLE_10RANKS | ADC group regular sequencer enable<br>with 10 ranks in the sequence                                             |
| LL_ADC_REG_SEQ_SCAN_ENABLE_11RANKS | ADC group regular sequencer enable<br>with 11 ranks in the sequence                                             |
| LL_ADC_REG_SEQ_SCAN_ENABLE_12RANKS | ADC group regular sequencer enable<br>with 12 ranks in the sequence                                             |
| LL_ADC_REG_SEQ_SCAN_ENABLE_13RANKS | ADC group regular sequencer enable<br>with 13 ranks in the sequence                                             |
| LL_ADC_REG_SEQ_SCAN_ENABLE_14RANKS | ADC group regular sequencer enable                                                                              |

|                                                  |                                                                                                                                                                                    |
|--------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                  | with 14 ranks in the sequence                                                                                                                                                      |
| LL_ADC_REG_SEQ_SCAN_ENABLE_15RANKS               | ADC group regular sequencer enable with 15 ranks in the sequence                                                                                                                   |
| LL_ADC_REG_SEQ_SCAN_ENABLE_16RANKS               | ADC group regular sequencer enable with 16 ranks in the sequence                                                                                                                   |
| <b><i>ADC group regular - Trigger edge</i></b>   |                                                                                                                                                                                    |
| LL_ADC_REG_TRIG_EXT_RISING                       | ADC group regular conversion trigger polarity set to rising edge                                                                                                                   |
| LL_ADC_REG_TRIG_EXT_FALLING                      | ADC group regular conversion trigger polarity set to falling edge                                                                                                                  |
| LL_ADC_REG_TRIG_EXT_RISINGFALLING                | ADC group regular conversion trigger polarity set to both rising and falling edges                                                                                                 |
| <b><i>ADC group regular - Trigger source</i></b> |                                                                                                                                                                                    |
| LL_ADC_REG_TRIG_SOFTWARE                         | ADC group regular conversion trigger internal: SW start.                                                                                                                           |
| LL_ADC_REG_TRIG_EXT_TIM1_CH1_ADC12               | ADC group regular conversion trigger from external IP: TIM1 channel 1 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT_TIM1_CH2_ADC12               | ADC group regular conversion trigger from external IP: TIM1 channel 2 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT_TIM1_CH3                     | ADC group regular conversion trigger from external IP: TIM1 channel 3 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT_TIM1_CH3                     | ADC group regular conversion trigger from external IP: TIM1 channel 3 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT_TIM2_CH2_ADC12               | ADC group regular conversion trigger from external IP: TIM2 channel 2 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT_TIM3_TRGO_ADC12              | ADC group regular conversion trigger from external IP: TIM3 TRGO. Trigger edge set to rising edge                                                                                  |

|                                       |                                                                                                                                                                                    |
|---------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                       | (default setting).                                                                                                                                                                 |
| LL_ADC_REG_TRIG_EXT_TIM4_CH4_ADC12    | ADC group regular conversion trigger from external IP: TIM4 channel 4 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT EXTI_LINE11_ADC12 | ADC group regular conversion trigger from external IP: external interrupt line 11. Trigger edge set to rising edge (default setting).                                              |
| LL_ADC_REG_TRIG_EXT_TIM8_TRGO_ADC12   | ADC group regular conversion trigger from external IP: TIM8 TRGO. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_REG_TRIG_EXT_TIM8_TRGO2        | ADC group regular conversion trigger from external IP: TIM8 TRGO2. Trigger edge set to rising edge (default setting).                                                              |
| LL_ADC_REG_TRIG_EXT_TIM8_TRGO2        | ADC group regular conversion trigger from external IP: TIM8 TRGO2. Trigger edge set to rising edge (default setting).                                                              |
| LL_ADC_REG_TRIG_EXT_TIM1_TRGO         | ADC group regular conversion trigger from external IP: TIM1 TRGO. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_REG_TRIG_EXT_TIM1_TRGO         | ADC group regular conversion trigger from external IP: TIM1 TRGO. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_REG_TRIG_EXT_TIM1_TRGO2        | ADC group regular conversion trigger from external IP: TIM1 TRGO2. Trigger edge set to rising edge (default setting).                                                              |
| LL_ADC_REG_TRIG_EXT_TIM1_TRGO2        | ADC group regular conversion trigger from external IP: TIM1 TRGO2. Trigger edge set to rising edge (default setting).                                                              |
| LL_ADC_REG_TRIG_EXT_TIM2_TRGO_ADC12   | ADC group regular conversion trigger from external IP: TIM2 TRGO. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_REG_TRIG_EXT_TIM4_TRGO         | ADC group regular conversion trigger from external IP: TIM4 TRGO. Trigger edge set to rising edge (default setting).                                                               |
| LL_ADC_REG_TRIG_EXT_TIM4_TRGO         | ADC group regular conversion trigger from external IP: TIM4 TRGO.                                                                                                                  |

|                                      |                                                                                                                                                                                    |
|--------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                      | Trigger edge set to rising edge<br>(default setting).                                                                                                                              |
| LL_ADC_REG_TRIG_EXT_TIM6_TRGO_ADC12  | ADC group regular conversion trigger from external IP: TIM6 TRGO.<br>Trigger edge set to rising edge<br>(default setting).                                                         |
| LL_ADC_REG_TRIG_EXT_TIM15_TRGO       | ADC group regular conversion trigger from external IP: TIM15 TRGO.<br>Trigger edge set to rising edge<br>(default setting).                                                        |
| LL_ADC_REG_TRIG_EXT_TIM15_TRGO       | ADC group regular conversion trigger from external IP: TIM15 TRGO.<br>Trigger edge set to rising edge<br>(default setting).                                                        |
| LL_ADC_REG_TRIG_EXT_TIM3_CH4_ADC12   | ADC group regular conversion trigger from external IP: TIM3 channel 4 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT_TIM3_CH1_ADC34   | ADC group regular conversion trigger from external IP: TIM3 channel 1 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT_TIM2_CH3_ADC34   | ADC group regular conversion trigger from external IP: TIM2 channel 3 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT_TIM8_CH1_ADC34   | ADC group regular conversion trigger from external IP: TIM8 channel 1 event (capture compare: input capture or output capture). Trigger edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT_TIM8_TRGO_ADC34  | ADC group regular conversion trigger from external IP: TIM8 TRGO.<br>Trigger edge set to rising edge<br>(default setting).                                                         |
| LL_ADC_REG_TRIG_EXT EXTI_LINE2_ADC34 | ADC group regular conversion trigger from external IP: external interrupt line 2. Trigger edge set to rising edge (default setting).                                               |
| LL_ADC_REG_TRIG_EXT_TIM4_CH1_ADC34   | ADC group regular conversion trigger from external IP: TIM4 channel 1 event (capture compare: input capture or output capture). Trigger                                            |

|                                     |                                                                                                                         |                                            |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------|--------------------------------------------|
|                                     |                                                                                                                         | edge set to rising edge (default setting). |
| LL_ADC_REG_TRIG_EXT_TIM2_TRGO_ADC34 | ADC group regular conversion trigger from external IP: TIM2 TRGO.<br>Trigger edge set to rising edge (default setting). |                                            |
| LL_ADC_REG_TRIG_EXT_TIM3_TRGO_ADC34 | ADC group regular conversion trigger from external IP: TIM3 TRGO.<br>Trigger edge set to rising edge (default setting). |                                            |
| LL_ADC_REG_TRIG_EXT_TIM7_TRGO_ADC34 | ADC group regular conversion trigger from external IP: TIM7 TRGO.<br>Trigger edge set to rising edge (default setting). |                                            |
| LL_ADC_REG_TRIG_EXT_TIM2_CC1_ADC34  | ADC group regular conversion trigger from external IP: TIM2 CCx. Trigger edge set to rising edge (default setting).     |                                            |

**ADC instance - Resolution**

|                       |                        |
|-----------------------|------------------------|
| LL_ADC_RESOLUTION_12B | ADC resolution 12 bits |
| LL_ADC_RESOLUTION_10B | ADC resolution 10 bits |
| LL_ADC_RESOLUTION_8B  | ADC resolution 8 bits  |
| LL_ADC_RESOLUTION_6B  | ADC resolution 6 bits  |

**ADC helper macro**

`_LL_ADC_CHANNEL_TO_DECIMAL_NB`

**Description:**

- Helper macro to get ADC channel number in decimal format from literals  
`LL_ADC_CHANNEL_x`.

**Parameters:**

- `_CHANNEL_`: This parameter can be one of the following values:
  - `LL_ADC_CHANNEL_0`
  - `LL_ADC_CHANNEL_1`
  - `LL_ADC_CHANNEL_2`
  - `LL_ADC_CHANNEL_3`
  - `LL_ADC_CHANNEL_4`
  - `LL_ADC_CHANNEL_5`
  - `LL_ADC_CHANNEL_6`
  - `LL_ADC_CHANNEL_7`
  - `LL_ADC_CHANNEL_8`
  - `LL_ADC_CHANNEL_9`
  - `LL_ADC_CHANNEL_10`
  - `LL_ADC_CHANNEL_11`
  - `LL_ADC_CHANNEL_12`
  - `LL_ADC_CHANNEL_13`
  - `LL_ADC_CHANNEL_14`
  - `LL_ADC_CHANNEL_15`

- LL\_ADC\_CHANNEL\_16
- LL\_ADC\_CHANNEL\_17
- LL\_ADC\_CHANNEL\_18
- LL\_ADC\_CHANNEL\_VREFINT (5)
- LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
- LL\_ADC\_CHANNEL\_VBAT (1)
- LL\_ADC\_CHANNEL\_VOPAMP1 (1)
- LL\_ADC\_CHANNEL\_VOPAMP2 (2)
- LL\_ADC\_CHANNEL\_VOPAMP3 (3)
- LL\_ADC\_CHANNEL\_VOPAMP4 (4)

**Return value:**

- Value: between Min\_Data=0 and Max\_Data=18

**Notes:**

- Example:  
`_LL_ADC_CHANNEL_TO_DECIMAL_NB(LL_ADC_CHANNEL_4)` will return decimal number "4".  
 The input can be a value from functions where a channel number is returned, either defined with number or with bitfield (only one bit must be set).

[\\_LL\\_ADC\\_DECIMAL\\_NB\\_TO\\_CHANNEL](#)

**Description:**

- Helper macro to get ADC channel in literal format `LL_ADC_CHANNEL_x` from number in decimal format.

**Parameters:**

- [\\_\\_DECIMAL\\_NB\\_\\_](#): Value between Min\_Data=0 and Max\_Data=18

**Return value:**

- Returned: value can be one of the following values:

- LL\_ADC\_CHANNEL\_0
- LL\_ADC\_CHANNEL\_1
- LL\_ADC\_CHANNEL\_2
- LL\_ADC\_CHANNEL\_3
- LL\_ADC\_CHANNEL\_4
- LL\_ADC\_CHANNEL\_5
- LL\_ADC\_CHANNEL\_6
- LL\_ADC\_CHANNEL\_7
- LL\_ADC\_CHANNEL\_8
- LL\_ADC\_CHANNEL\_9
- LL\_ADC\_CHANNEL\_10
- LL\_ADC\_CHANNEL\_11
- LL\_ADC\_CHANNEL\_12
- LL\_ADC\_CHANNEL\_13
- LL\_ADC\_CHANNEL\_14
- LL\_ADC\_CHANNEL\_15
- LL\_ADC\_CHANNEL\_16
- LL\_ADC\_CHANNEL\_17
- LL\_ADC\_CHANNEL\_18

- LL\_ADC\_CHANNEL\_VREFINT (5)
- LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
- LL\_ADC\_CHANNEL\_VBAT (1)
- LL\_ADC\_CHANNEL\_VOPAMP1 (1)
- LL\_ADC\_CHANNEL\_VOPAMP2 (2)
- LL\_ADC\_CHANNEL\_VOPAMP3 (3)
- LL\_ADC\_CHANNEL\_VOPAMP4 (4)

**Notes:**

- Example:  
`__LL_ADC_DECIMAL_NB_TO_CHANNEL(4)` will return a data equivalent to "LL\_ADC\_CHANNEL\_4".

LL\_ADC\_IS\_CHANNEL\_INTERNAL

**Description:**

- Helper macro to determine whether the selected channel corresponds to literal definitions of driver.

**Parameters:**

- CHANNEL: This parameter can be one of the following values:
  - LL\_ADC\_CHANNEL\_0
  - LL\_ADC\_CHANNEL\_1
  - LL\_ADC\_CHANNEL\_2
  - LL\_ADC\_CHANNEL\_3
  - LL\_ADC\_CHANNEL\_4
  - LL\_ADC\_CHANNEL\_5
  - LL\_ADC\_CHANNEL\_6
  - LL\_ADC\_CHANNEL\_7
  - LL\_ADC\_CHANNEL\_8
  - LL\_ADC\_CHANNEL\_9
  - LL\_ADC\_CHANNEL\_10
  - LL\_ADC\_CHANNEL\_11
  - LL\_ADC\_CHANNEL\_12
  - LL\_ADC\_CHANNEL\_13
  - LL\_ADC\_CHANNEL\_14
  - LL\_ADC\_CHANNEL\_15
  - LL\_ADC\_CHANNEL\_16
  - LL\_ADC\_CHANNEL\_17
  - LL\_ADC\_CHANNEL\_18
  - LL\_ADC\_CHANNEL\_VREFINT (5)
  - LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
  - LL\_ADC\_CHANNEL\_VBAT (1)
  - LL\_ADC\_CHANNEL\_VOPAMP1 (1)
  - LL\_ADC\_CHANNEL\_VOPAMP2 (2)
  - LL\_ADC\_CHANNEL\_VOPAMP3 (3)
  - LL\_ADC\_CHANNEL\_VOPAMP4 (4)

**Return value:**

- Value: "0" if the channel corresponds to a parameter definition of a ADC external channel (channel connected to a GPIO pin). Value "1" if the channel corresponds to a parameter definition

of a ADC internal channel.

**Notes:**

- The different literal definitions of ADC channels are: ADC internal channel:  
LL\_ADC\_CHANNEL\_VREFINT,  
LL\_ADC\_CHANNEL\_TEMPSENSOR, ...ADC external channel (channel connected to a GPIO pin): LL\_ADC\_CHANNEL\_1,  
LL\_ADC\_CHANNEL\_2, ... The channel parameter must be a value defined from literal definition of a ADC internal channel  
(LL\_ADC\_CHANNEL\_VREFINT,  
LL\_ADC\_CHANNEL\_TEMPSENSOR, ...), ADC external channel (LL\_ADC\_CHANNEL\_1,  
LL\_ADC\_CHANNEL\_2, ...), must not be a value from functions where a channel number is returned from ADC registers, because internal and external channels share the same channel number in ADC registers. The differentiation is made only with parameters definitions of driver.

LL\_ADC\_CHANNEL\_INTERNAL\_TO\_EXTERNAL

**Description:**

- Helper macro to convert a channel defined from parameter definition of a ADC internal channel (LL\_ADC\_CHANNEL\_VREFINT, LL\_ADC\_CHANNEL\_TEMPSENSOR, ...), to its equivalent parameter definition of a ADC external channel (LL\_ADC\_CHANNEL\_1, LL\_ADC\_CHANNEL\_2, ...).

**Parameters:**

- \_\_CHANNEL\_\_: This parameter can be one of the following values:
  - LL\_ADC\_CHANNEL\_0
  - LL\_ADC\_CHANNEL\_1
  - LL\_ADC\_CHANNEL\_2
  - LL\_ADC\_CHANNEL\_3
  - LL\_ADC\_CHANNEL\_4
  - LL\_ADC\_CHANNEL\_5
  - LL\_ADC\_CHANNEL\_6
  - LL\_ADC\_CHANNEL\_7
  - LL\_ADC\_CHANNEL\_8
  - LL\_ADC\_CHANNEL\_9
  - LL\_ADC\_CHANNEL\_10
  - LL\_ADC\_CHANNEL\_11
  - LL\_ADC\_CHANNEL\_12
  - LL\_ADC\_CHANNEL\_13
  - LL\_ADC\_CHANNEL\_14
  - LL\_ADC\_CHANNEL\_15
  - LL\_ADC\_CHANNEL\_16
  - LL\_ADC\_CHANNEL\_17
  - LL\_ADC\_CHANNEL\_18
  - LL\_ADC\_CHANNEL\_VREFINT (5)



- LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
- LL\_ADC\_CHANNEL\_VBAT (1)
- LL\_ADC\_CHANNEL\_VOPAMP1 (1)
- LL\_ADC\_CHANNEL\_VOPAMP2 (2)
- LL\_ADC\_CHANNEL\_VOPAMP3 (3)
- LL\_ADC\_CHANNEL\_VOPAMP4 (4)

**Return value:**

- Returned: value can be one of the following values:
  - LL\_ADC\_CHANNEL\_0
  - LL\_ADC\_CHANNEL\_1
  - LL\_ADC\_CHANNEL\_2
  - LL\_ADC\_CHANNEL\_3
  - LL\_ADC\_CHANNEL\_4
  - LL\_ADC\_CHANNEL\_5
  - LL\_ADC\_CHANNEL\_6
  - LL\_ADC\_CHANNEL\_7
  - LL\_ADC\_CHANNEL\_8
  - LL\_ADC\_CHANNEL\_9
  - LL\_ADC\_CHANNEL\_10
  - LL\_ADC\_CHANNEL\_11
  - LL\_ADC\_CHANNEL\_12
  - LL\_ADC\_CHANNEL\_13
  - LL\_ADC\_CHANNEL\_14
  - LL\_ADC\_CHANNEL\_15
  - LL\_ADC\_CHANNEL\_16
  - LL\_ADC\_CHANNEL\_17
  - LL\_ADC\_CHANNEL\_18

**Notes:**

- The channel parameter can be, additionally to a value defined from parameter definition of a ADC internal channel (LL\_ADC\_CHANNEL\_VREFINT, LL\_ADC\_CHANNEL\_TEMPSENSOR, ...), a value defined from parameter definition of ADC external channel (LL\_ADC\_CHANNEL\_1, LL\_ADC\_CHANNEL\_2, ...) or a value from functions where a channel number is returned from ADC registers.

`_LL_ADC_IS_CHANNEL_INTERNAL_AVAILABLE`**Description:**

- Helper macro to determine whether the internal channel selected is available on the ADC instance selected.

**Parameters:**

- `_ADC_INSTANCE_`: ADC instance
- `_CHANNEL_`: This parameter can be one of the following values:
  - `LL_ADC_CHANNEL_VREFINT` (5)
  - `LL_ADC_CHANNEL_TEMPSENSOR` (1)
  - `LL_ADC_CHANNEL_VBAT` (1)
  - `LL_ADC_CHANNEL_VOPAMP1` (1)
  - `LL_ADC_CHANNEL_VOPAMP2` (2)
  - `LL_ADC_CHANNEL_VOPAMP3` (3)
  - `LL_ADC_CHANNEL_VOPAMP4` (4)

**Return value:**

- Value: "0" if the internal channel selected is not available on the ADC instance selected. Value "1" if the internal channel selected is available on the ADC instance selected.

**Notes:**

- The channel parameter must be a value defined from parameter definition of a ADC internal channel (`LL_ADC_CHANNEL_VREFINT`, `LL_ADC_CHANNEL_TEMPSENSOR`, ...), must not be a value defined from parameter definition of ADC external channel (`LL_ADC_CHANNEL_1`, `LL_ADC_CHANNEL_2`, ...) or a value from functions where a channel number is returned from ADC registers, because internal and external channels share the same channel number in ADC registers. The differentiation is made only with parameters definitions of driver.

`_LL_ADC_ANALOGWD_CHANNEL_GROUP`**Description:**

- Helper macro to define ADC analog watchdog parameter: define a single channel to monitor with analog watchdog from sequencer channel and groups definition.

**Parameters:**

- `_CHANNEL_`: This parameter can be one of the following values:
  - `LL_ADC_CHANNEL_0`
  - `LL_ADC_CHANNEL_1`
  - `LL_ADC_CHANNEL_2`
  - `LL_ADC_CHANNEL_3`
  - `LL_ADC_CHANNEL_4`
  - `LL_ADC_CHANNEL_5`
  - `LL_ADC_CHANNEL_6`

- LL\_ADC\_CHANNEL\_7
- LL\_ADC\_CHANNEL\_8
- LL\_ADC\_CHANNEL\_9
- LL\_ADC\_CHANNEL\_10
- LL\_ADC\_CHANNEL\_11
- LL\_ADC\_CHANNEL\_12
- LL\_ADC\_CHANNEL\_13
- LL\_ADC\_CHANNEL\_14
- LL\_ADC\_CHANNEL\_15
- LL\_ADC\_CHANNEL\_16
- LL\_ADC\_CHANNEL\_17
- LL\_ADC\_CHANNEL\_18
- LL\_ADC\_CHANNEL\_VREFINT (5)
- LL\_ADC\_CHANNEL\_TEMPSENSOR (1)
- LL\_ADC\_CHANNEL\_VBAT (1)
- LL\_ADC\_CHANNEL\_VOPAMP1 (1)
- LL\_ADC\_CHANNEL\_VOPAMP2 (2)
- LL\_ADC\_CHANNEL\_VOPAMP3 (3)
- LL\_ADC\_CHANNEL\_VOPAMP4 (4)
- \_\_GROUP\_\_: This parameter can be one of the following values:
  - LL\_ADC\_GROUP\_REGULAR
  - LL\_ADC\_GROUP\_INJECTED
  - LL\_ADC\_GROUP\_REGULAR\_INJECTED

**Return value:**

- Returned: value can be one of the following values:
  - LL\_ADC\_AWD\_DISABLE
  - LL\_ADC\_AWD\_ALL\_CHANNELS\_REG (0)
  - LL\_ADC\_AWD\_ALL\_CHANNELS\_INJ (0)
  - LL\_ADC\_AWD\_ALL\_CHANNELS\_REG\_INJ
  - LL\_ADC\_AWD\_CHANNEL\_0\_REG (0)
  - LL\_ADC\_AWD\_CHANNEL\_0\_INJ (0)
  - LL\_ADC\_AWD\_CHANNEL\_0\_REG\_INJ
  - LL\_ADC\_AWD\_CHANNEL\_1\_REG (0)
  - LL\_ADC\_AWD\_CHANNEL\_1\_INJ (0)
  - LL\_ADC\_AWD\_CHANNEL\_1\_REG\_INJ
  - LL\_ADC\_AWD\_CHANNEL\_2\_REG (0)
  - LL\_ADC\_AWD\_CHANNEL\_2\_INJ (0)
  - LL\_ADC\_AWD\_CHANNEL\_2\_REG\_INJ
  - LL\_ADC\_AWD\_CHANNEL\_3\_REG (0)
  - LL\_ADC\_AWD\_CHANNEL\_3\_INJ (0)
  - LL\_ADC\_AWD\_CHANNEL\_3\_REG\_INJ
  - LL\_ADC\_AWD\_CHANNEL\_4\_REG (0)
  - LL\_ADC\_AWD\_CHANNEL\_4\_INJ (0)
  - LL\_ADC\_AWD\_CHANNEL\_4\_REG\_INJ
  - LL\_ADC\_AWD\_CHANNEL\_5\_REG (0)
  - LL\_ADC\_AWD\_CHANNEL\_5\_INJ (0)
  - LL\_ADC\_AWD\_CHANNEL\_5\_REG\_INJ
  - LL\_ADC\_AWD\_CHANNEL\_6\_REG (0)
  - LL\_ADC\_AWD\_CHANNEL\_6\_INJ (0)

- LL\_ADC\_AWD\_CHANNEL\_6\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_7\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_7\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_7\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_8\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_8\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_8\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_9\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_9\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_9\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_10\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_10\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_10\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_11\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_11\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_11\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_12\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_12\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_12\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_13\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_13\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_13\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_14\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_14\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_14\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_15\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_15\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_15\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_16\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_16\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_16\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_17\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_17\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_17\_REG\_INJ
- LL\_ADC\_AWD\_CHANNEL\_18\_REG (0)
- LL\_ADC\_AWD\_CHANNEL\_18\_INJ (0)
- LL\_ADC\_AWD\_CHANNEL\_18\_REG\_INJ
- LL\_ADC\_AWD\_CH\_VREFINT\_REG (0)(5)
- LL\_ADC\_AWD\_CH\_VREFINT\_INJ (0)(5)
- LL\_ADC\_AWD\_CH\_VREFINT\_REG\_INJ (5)
- LL\_ADC\_AWD\_CH\_TEMPSENSOR\_REG (0)(1)
- LL\_ADC\_AWD\_CH\_TEMPSENSOR\_INJ (0)(1)
- LL\_ADC\_AWD\_CH\_TEMPSENSOR\_REG\_I NJ (1)
- LL\_ADC\_AWD\_CH\_VBAT\_REG (0)(1)
- LL\_ADC\_AWD\_CH\_VBAT\_INJ (0)(1)
- LL\_ADC\_AWD\_CH\_VBAT\_REG\_INJ (1)
- LL\_ADC\_AWD\_CH\_VOPAMP1\_REG (0)(1)
- LL\_ADC\_AWD\_CH\_VOPAMP1\_INJ (0)(1)
- LL\_ADC\_AWD\_CH\_VOPAMP1\_REG\_INJ (1)

- LL\_ADC\_AWD\_CH\_VOPAMP2\_REG (0)(2)
- LL\_ADC\_AWD\_CH\_VOPAMP2\_INJ (0)(2)
- LL\_ADC\_AWD\_CH\_VOPAMP2\_REG\_INJ (2)
- LL\_ADC\_AWD\_CH\_VOPAMP3\_REG (0)(3)
- LL\_ADC\_AWD\_CH\_VOPAMP3\_INJ (0)(3)
- LL\_ADC\_AWD\_CH\_VOPAMP3\_REG\_INJ (3)
- LL\_ADC\_AWD\_CH\_VOPAMP4\_REG (0)(4)
- LL\_ADC\_AWD\_CH\_VOPAMP4\_INJ (0)(4)
- LL\_ADC\_AWD\_CH\_VOPAMP4\_REG\_INJ (4)

**Notes:**

- To be used with function  
`LL_ADC_SetAnalogWDMonitChannels()`.  
Example: `LL_ADC_SetAnalogWDMonitChannels(ADC1, LL_ADC_AWD1,  
__LL_ADC_ANALOGWD_CHANNEL_GROUP(LL_ADC_CHANNEL4,  
LL_ADC_GROUP_REGULAR))`

[\\_\\_LL\\_ADC\\_ANALOGWD\\_SET\\_THRESHOLD\\_RESOLUTION](#)**Description:**

- Helper macro to set the value of ADC analog watchdog threshold high or low in function of ADC resolution, when ADC resolution is different of 12 bits.

**Parameters:**

- \_\_ADC\_RESOLUTION\_\_: This parameter can be one of the following values:
  - LL\_ADC\_RESOLUTION\_12B
  - LL\_ADC\_RESOLUTION\_10B
  - LL\_ADC\_RESOLUTION\_8B
  - LL\_ADC\_RESOLUTION\_6B
- \_\_AWD\_THRESHOLD\_\_: Value between Min\_Data=0x000 and Max\_Data=0xFFFF

**Return value:**

- Value: between Min\_Data=0x000 and Max\_Data=0xFFFF

**Notes:**

- To be used with function  
`LL_ADC_ConfigAnalogWDThresholds()` or  
`LL_ADC_SetAnalogWDThresholds()`. Example,  
with a ADC resolution of 8 bits, to set the value of  
analog watchdog threshold high (on 8 bits):  
`LL_ADC_SetAnalogWDThresholds (< ADCx param>,  
__LL_ADC_ANALOGWD_SET_THRESHOLD_RESOLUTION(LL_ADC_RESOLUTION_8B,  
<threshold_value_8_bits>) );`

[\\_\\_LL\\_ADC\\_ANALOGWD\\_GET\\_THRESHOLD\\_RESOLUTION](#)**Description:**

- Helper macro to get the value of ADC analog watchdog threshold high or low in function of ADC resolution, when ADC resolution is different of 12 bits.

**Parameters:**

- [\\_\\_ADC\\_RESOLUTION\\_\\_](#): This parameter can be one of the following values:
  - [LL\\_ADC\\_RESOLUTION\\_12B](#)
  - [LL\\_ADC\\_RESOLUTION\\_10B](#)
  - [LL\\_ADC\\_RESOLUTION\\_8B](#)
  - [LL\\_ADC\\_RESOLUTION\\_6B](#)
- [\\_\\_AWD\\_THRESHOLD\\_12\\_BITS\\_\\_](#): Value between Min\_Data=0x000 and Max\_Data=0xFFFF

**Return value:**

- Value: between Min\_Data=0x000 and Max\_Data=0xFFFF

**Notes:**

- To be used with function LL\_ADC\_GetAnalogWDThresholds(). Example, with a ADC resolution of 8 bits, to get the value of analog watchdog threshold high (on 8 bits): <threshold\_value\_6\_bits> = [\\_\\_LL\\_ADC\\_ANALOGWD\\_GET\\_THRESHOLD\\_RESOLUTION\(LL\\_ADC\\_RESOLUTION\\_8B, LL\\_ADC\\_GetAnalogWDThresholds\(<ADCx param>, LL\\_ADC\\_AWD\\_THRESHOLD\\_HIGH\)\)](#);

[\\_\\_LL\\_ADC\\_ANALOGWD\\_THRESHOLDS\\_HIGH\\_LOW](#)**Description:**

- Helper macro to get the ADC analog watchdog threshold high or low from raw value containing both thresholds concatenated.

**Parameters:**

- [\\_\\_AWD\\_THRESHOLD\\_TYPE\\_\\_](#): This parameter can be one of the following values:
  - [LL\\_ADC\\_AWD\\_THRESHOLD\\_HIGH](#)
  - [LL\\_ADC\\_AWD\\_THRESHOLD\\_LOW](#)
- [\\_\\_AWD\\_THRESHOLDS\\_\\_](#): Value between Min\_Data=0x00000000 and Max\_Data=0xFFFFFFFF

**Return value:**

- Value: between Min\_Data=0x000 and Max\_Data=0xFFFF

**Notes:**

- To be used with function LL\_ADC\_GetAnalogWDThresholds(). Example, to

---

get analog watchdog threshold high from the register raw value:  
`__LL_ADC_ANALOGWD_THRESHOLDS_HIGH_LOW(LL_ADC_AWD_THRESHOLD_HIGH,  
<raw_value_with_both_thresholds>);`

### `__LL_ADC_CALIB_FACTOR_SINGLE_DIFF`

**Description:**

- Helper macro to set the ADC calibration value with both single ended and differential modes calibration factors concatenated.

**Parameters:**

- `__CALIB_FACTOR_SINGLE_ENDED__`: Value between Min\_Data=0x00 and Max\_Data=0x7F
- `__CALIB_FACTOR_DIFFERENTIAL__`: Value between Min\_Data=0x00 and Max\_Data=0x7F

**Return value:**

- Value: between Min\_Data=0x00000000 and Max\_Data=0xFFFFFFFF

**Notes:**

- To be used with function `LL_ADC_SetCalibrationFactor()`. Example, to set calibration factors single ended to 0x55 and differential ended to 0x2A:  
`LL_ADC_SetCalibrationFactor( ADC1,  
__LL_ADC_CALIB_FACTOR_SINGLE_DIFF(0x55, 0x2A))`

### `__LL_ADC_MULTI_CONV_DATA_MASTER_SLAVE`

**Description:**

- Helper macro to get the ADC multimode conversion data of ADC master or ADC slave from raw value with both ADC conversion data concatenated.

**Parameters:**

- `__ADC_MULTI_MASTER_SLAVE__`: This parameter can be one of the following values:
  - `LL_ADC_MULTI_MASTER`
  - `LL_ADC_MULTI_SLAVE`
- `__ADC_MULTI_CONV_DATA__`: Value between Min\_Data=0x000 and Max\_Data=0xFFFF

**Return value:**

- Value: between Min\_Data=0x000 and Max\_Data=0xFFFF

**Notes:**

- This macro is intended to be used when multimode transfer by DMA is enabled: refer to function `LL_ADC_SetMultiDMATransfer()`. In this case the transferred data need to be processed with this macro to separate the conversion data of ADC

master and ADC slave.

### \_\_LL\_ADC\_COMMON\_INSTANCE

#### Description:

- Helper macro to select the ADC common instance to which is belonging the selected ADC instance.

#### Parameters:

- \_\_ADCx\_\_: ADC instance

#### Return value:

- ADC: common register instance

#### Notes:

- ADC common register instance can be used for: Set parameters common to several ADC instancesMultimode (for devices with several ADC instances) Refer to functions having argument "ADCxy\_COMMON" as parameter.

### \_\_LL\_ADC\_IS\_ENABLED\_ALL\_COMMON\_INSTANCE

#### Description:

- Helper macro to check if all ADC instances sharing the same ADC common instance are disabled.

#### Parameters:

- \_\_ADCXY\_COMMON\_\_: ADC common instance (can be set directly from CMSIS definition or by using helper macro

#### Return value:

- Value: "0" if all ADC instances sharing the same ADC common instance are disabled. Value "1" if at least one ADC instance sharing the same ADC common instance is enabled.

#### Notes:

- This check is required by functions with setting conditioned to ADC state: All ADC instances of the ADC common group must be disabled. Refer to functions having argument "ADCxy\_COMMON" as parameter. On devices with only 1 ADC common instance, parameter of this macro is useless and can be ignored (parameter kept for compatibility with devices featuring several ADC common instances).

### \_\_LL\_ADC\_DIGITAL\_SCALE

#### Description:

- Helper macro to define the ADC conversion data full-scale digital value corresponding to the selected ADC resolution.

#### Parameters:

- \_\_ADC\_RESOLUTION\_\_: This parameter can be one of the following values:

- LL\_ADC\_RESOLUTION\_12B
- LL\_ADC\_RESOLUTION\_10B
- LL\_ADC\_RESOLUTION\_8B
- LL\_ADC\_RESOLUTION\_6B

**Return value:**

- ADC: conversion data equivalent voltage value (unit: mVolt)

**Notes:**

- ADC conversion data full-scale corresponds to voltage range determined by analog voltage references Vref+ and Vref- (refer to reference manual).

**\_\_LL\_ADC\_CONVERT\_DATA\_RESOLUTION****Description:**

- Helper macro to convert the ADC conversion data from a resolution to another resolution.

**Parameters:**

- \_\_DATA\_\_: ADC conversion data to be converted
- \_\_ADC\_RESOLUTION\_CURRENT\_\_: Resolution of the data to be converted This parameter can be one of the following values:
  - LL\_ADC\_RESOLUTION\_12B
  - LL\_ADC\_RESOLUTION\_10B
  - LL\_ADC\_RESOLUTION\_8B
  - LL\_ADC\_RESOLUTION\_6B
- \_\_ADC\_RESOLUTION\_TARGET\_\_: Resolution of the data after conversion This parameter can be one of the following values:
  - LL\_ADC\_RESOLUTION\_12B
  - LL\_ADC\_RESOLUTION\_10B
  - LL\_ADC\_RESOLUTION\_8B
  - LL\_ADC\_RESOLUTION\_6B

**Return value:**

- ADC: conversion data to the requested resolution

**\_\_LL\_ADC\_CALC\_DATA\_TO\_VOLTAGE****Description:**

- Helper macro to calculate the voltage (unit: mVolt) corresponding to a ADC conversion data (unit: digital value).

**Parameters:**

- \_\_VREFANALOG\_VOLTAGE\_\_: Analog reference voltage (unit: mV)
- \_\_ADC\_DATA\_\_: ADC conversion data (resolution 12 bits) (unit: digital value).
- \_\_ADC\_RESOLUTION\_\_: This parameter can be one of the following values:
  - LL\_ADC\_RESOLUTION\_12B
  - LL\_ADC\_RESOLUTION\_10B
  - LL\_ADC\_RESOLUTION\_8B

- LL\_ADC\_RESOLUTION\_6B

**Return value:**

- ADC: conversion data equivalent voltage value  
(unit: mVolt)

**Notes:**

- Analog reference voltage (Vref+) must be either known from user board environment or can be calculated using ADC measurement and ADC helper macro  
\_\_LL\_ADC\_CALC\_VREFANALOG\_VOLTAGE().

\_\_LL\_ADC\_CALC\_VREF  
ANALOG\_VOLTAGE

**Description:**

- Helper macro to calculate analog reference voltage (Vref+) (unit: mVolt) from ADC conversion data of internal voltage reference VrefInt.

**Parameters:**

- \_\_VREFINT\_ADC\_DATA\_\_: ADC conversion data (resolution 12 bits) of internal voltage reference VrefInt (unit: digital value).
- \_\_ADC\_RESOLUTION\_\_: This parameter can be one of the following values:
  - LL\_ADC\_RESOLUTION\_12B
  - LL\_ADC\_RESOLUTION\_10B
  - LL\_ADC\_RESOLUTION\_8B
  - LL\_ADC\_RESOLUTION\_6B

**Return value:**

- Analog: reference voltage (unit: mV)

**Notes:**

- Computation is using VrefInt calibration value stored in system memory for each device during production. This voltage depends on user board environment: voltage level connected to pin Vref+. On devices with small package, the pin Vref+ is not present and internally bonded to pin Vdda. On this STM32 serie, calibration data of internal voltage reference VrefInt corresponds to a resolution of 12 bits, this is the recommended ADC resolution to convert voltage of internal voltage reference VrefInt. Otherwise, this macro performs the processing to scale ADC conversion data to 12 bits.

\_\_LL\_ADC\_CALC\_  
TEMPERATURE

**Description:**

- Helper macro to calculate the temperature (unit: degree Celsius) from ADC conversion data of internal temperature sensor.

**Parameters:**

- \_\_VREFANALOG\_VOLTAGE\_\_: Analog

- reference voltage (unit: mV)
- `__TEMPSENSOR_ADC_DATA__`: ADC conversion data of internal temperature sensor (unit: digital value).
- `__ADC_RESOLUTION__`: ADC resolution at which internal temperature sensor voltage has been measured. This parameter can be one of the following values:
  - `LL_ADC_RESOLUTION_12B`
  - `LL_ADC_RESOLUTION_10B`
  - `LL_ADC_RESOLUTION_8B`
  - `LL_ADC_RESOLUTION_6B`

**Return value:**

- Temperature: (unit: degree Celsius)

**Notes:**

- Computation is using temperature sensor calibration values stored in system memory for each device during production. Calculation formula: Temperature = ((TS\_ADC\_DATA - TS\_CAL1) \* (TS\_CAL2\_TEMP - TS\_CAL1\_TEMP)) / (TS\_CAL2 - TS\_CAL1) + TS\_CAL1\_TEMP with TS\_ADC\_DATA = temperature sensor raw data measured by ADC Avg\_Slope = (TS\_CAL2 - TS\_CAL1) / (TS\_CAL2\_TEMP - TS\_CAL1\_TEMP) TS\_CAL1 = equivalent TS\_ADC\_DATA at temperature TEMP\_DEGC\_CAL1 (calibrated in factory) TS\_CAL2 = equivalent TS\_ADC\_DATA at temperature TEMP\_DEGC\_CAL2 (calibrated in factory) Caution: Calculation relevancy under reserve that calibration parameters are correct (address and data). To calculate temperature using temperature sensor datasheet typical values (generic values less, therefore less accurate than calibrated values), use helper macro `__LL_ADC_CALC_TEMPERATURE_TYP_PARA_MS()`. As calculation input, the analog reference voltage (Vref+) must be defined as it impacts the ADC LSB equivalent voltage. Analog reference voltage (Vref+) must be either known from user board environment or can be calculated using ADC measurement and ADC helper macro `__LL_ADC_CALC_VREFANALOG_VOLTAGE()`. On this STM32 serie, calibration data of temperature sensor corresponds to a resolution of 12 bits, this is the recommended ADC resolution to convert voltage of temperature sensor. Otherwise, this macro performs the processing to scale ADC conversion data to 12 bits.

[\\_\\_LL\\_ADC\\_CALC\\_TEMPERATURE\\_TYP\\_PARAMS](#)**Description:**

- Helper macro to calculate the temperature (unit: degree Celsius) from ADC conversion data of internal temperature sensor.

**Parameters:**

- [\\_\\_TEMPSENSOR\\_TYP\\_AVGSLOPE](#): Device datasheet data: Temperature sensor slope typical value (unit: uV/DegCelsius). On STM32F3, refer to device datasheet parameter "Avg\_Slope".
- [\\_\\_TEMPSENSOR\\_TYP\\_CALX\\_V](#): Device datasheet data: Temperature sensor voltage typical value (at temperature and Vref+ defined in parameters below) (unit: mV). On STM32F3, refer to device datasheet parameter "V25" (corresponding to TS\_CAL1).
- [\\_\\_TEMPSENSOR\\_CALX\\_TEMP](#): Device datasheet data: Temperature at which temperature sensor voltage (see parameter above) is corresponding (unit: mV)
- [\\_\\_VREFANALOG\\_VOLTAGE](#): Analog voltage reference (Vref+) voltage (unit: mV)
- [\\_\\_TEMPSENSOR\\_ADC\\_DATA](#): ADC conversion data of internal temperature sensor (unit: digital value).
- [\\_\\_ADC\\_RESOLUTION](#): ADC resolution at which internal temperature sensor voltage has been measured. This parameter can be one of the following values:
  - [LL\\_ADC\\_RESOLUTION\\_12B](#)
  - [LL\\_ADC\\_RESOLUTION\\_10B](#)
  - [LL\\_ADC\\_RESOLUTION\\_8B](#)
  - [LL\\_ADC\\_RESOLUTION\\_6B](#)

**Return value:**

- Temperature: (unit: degree Celsius)

**Notes:**

- Computation is using temperature sensor typical values (refer to device datasheet). Calculation formula: Temperature =  $(TS\_TYP\_CALx\_VOLT(uV) - TS\_ADC\_DATA * Conversion\_uV) / Avg\_Slope + CALx\_TEMP$  with TS\_ADC\_DATA = temperature sensor raw data measured by ADC (unit: digital value) Avg\_Slope = temperature sensor slope (unit: uV/Degree Celsius) TS\_TYP\_CALx\_VOLT = temperature sensor digital value at temperature CALx\_TEMP (unit: mV) Caution: Calculation relevancy under reserve the temperature sensor of the current device has characteristics in line with datasheet typical values. If temperature sensor calibration values are available on this device (presence of macro [\\_\\_LL\\_ADC\\_CALC\\_TEMPERATURE\(\)](#)),

temperature calculation will be more accurate using helper macro

`__LL_ADC_CALC_TEMPERATURE()`. As calculation input, the analog reference voltage ( $V_{ref+}$ ) must be defined as it impacts the ADC LSB equivalent voltage. Analog reference voltage ( $V_{ref+}$ ) must be either known from user board environment or can be calculated using ADC measurement and ADC helper macro

`__LL_ADC_CALC_VREFANALOG_VOLTAGE()`. ADC measurement data must correspond to a resolution of 12bits (full scale digital value 4095). If not the case, the data must be preliminarily rescaled to an equivalent resolution of 12 bits.

#### **Common write and read registers Macros**

##### **LL\_ADC\_WriteReg**

###### **Description:**

- Write a value in ADC register.

###### **Parameters:**

- `__INSTANCE__`: ADC Instance
- `__REG__`: Register to be written
- `__VALUE__`: Value to be written in the register

###### **Return value:**

- None

##### **LL\_ADC\_ReadReg**

###### **Description:**

- Read a value in ADC register.

###### **Parameters:**

- `__INSTANCE__`: ADC Instance
- `__REG__`: Register to be read

###### **Return value:**

- Register: value

## 60 LL BUS Generic Driver

### 60.1 BUS Firmware driver API description

#### 60.1.1 Detailed description of functions

##### **LL\_AHB1\_GRP1\_EnableClock**

Function name **STATIC\_INLINE void LL\_AHB1\_GRP1\_EnableClock (uint32\_t Periph)**

Function description Enable AHB1 peripherals clock.

Parameters • **Periph:** This parameter can be a combination of the following values: (\*) value not defined in all devices.

- LL\_AHB1\_GRP1\_PERIPH\_DMA1
- LL\_AHB1\_GRP1\_PERIPH\_DMA2 (\*)
- LL\_AHB1\_GRP1\_PERIPH\_SRAM
- LL\_AHB1\_GRP1\_PERIPH\_FLASH
- LL\_AHB1\_GRP1\_PERIPH\_FMC (\*)
- LL\_AHB1\_GRP1\_PERIPH\_CRC
- LL\_AHB1\_GRP1\_PERIPH\_GPIOH (\*)
- LL\_AHB1\_GRP1\_PERIPH\_GPIOA
- LL\_AHB1\_GRP1\_PERIPH\_GPIOB
- LL\_AHB1\_GRP1\_PERIPH\_GPIOC
- LL\_AHB1\_GRP1\_PERIPH\_GPIOD
- LL\_AHB1\_GRP1\_PERIPH\_GPIOE (\*)
- LL\_AHB1\_GRP1\_PERIPH\_GPIOF
- LL\_AHB1\_GRP1\_PERIPH\_GPIOG (\*)
- LL\_AHB1\_GRP1\_PERIPH\_TSC
- LL\_AHB1\_GRP1\_PERIPH\_ADC1 (\*)
- LL\_AHB1\_GRP1\_PERIPH\_ADC12 (\*)
- LL\_AHB1\_GRP1\_PERIPH\_ADC34 (\*)

Return values • **None**

Reference Manual to LL API cross reference:  
 • AHBENR DMA1EN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR DMA2EN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR SRAMEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR FLITFEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR FMCEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR CRCEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR GPIOHEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR GPIOAEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR GPIOBEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR GPIOCEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR GPIODEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR GPIOEEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR GPIOFEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR GPIOGEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR TSCEN LL\_AHB1\_GRP1\_EnableClock  
 • AHBENR ADC1EN LL\_AHB1\_GRP1\_EnableClock

- AHBENR ADC12EN LL\_AHB1\_GRP1\_EnableClock
- AHBENR ADC34EN LL\_AHB1\_GRP1\_EnableClock

### LL\_AHB1\_GRP1\_IsEnabledClock

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_AHB1_GRP1_IsEnabledClock<br/>(uint32_t Periph)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description                              | Check if AHB1 peripheral clock is enabled or not.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- LL_AHB1_GRP1_PERIPH_DMA1</li> <li>- LL_AHB1_GRP1_PERIPH_DMA2 (*)</li> <li>- LL_AHB1_GRP1_PERIPH_SRAM</li> <li>- LL_AHB1_GRP1_PERIPH_FLASH</li> <li>- LL_AHB1_GRP1_PERIPH_FMC (*)</li> <li>- LL_AHB1_GRP1_PERIPH_CRC</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOH (*)</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOA</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOB</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOC</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOD</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOE (*)</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOF</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOG (*)</li> <li>- LL_AHB1_GRP1_PERIPH_TSC</li> <li>- LL_AHB1_GRP1_PERIPH_ADC1 (*)</li> <li>- LL_AHB1_GRP1_PERIPH_ADC12 (*)</li> <li>- LL_AHB1_GRP1_PERIPH_ADC34 (*)</li> </ul> </li> </ul>                                                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of Periph (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• AHBENR DMA1EN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR DMA2EN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR SRAMEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR FLITFEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR FMCEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR CRCEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR GPIOHEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR GPIOAEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR GPIOBEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR GPIOCEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR GPIODEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR GPIOEEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR GPIOFEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR GPIOGEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR TSCEN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR ADC1EN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR ADC12EN LL_AHB1_GRP1_IsEnabledClock</li> <li>• AHBENR ADC34EN LL_AHB1_GRP1_IsEnabledClock</li> </ul> |

**LL\_AHB1\_GRP1\_DisableClock**

Function name      **`_STATIC_INLINE void LL_AHB1_GRP1_DisableClock  
(uint32_t Periph)`**

Function description      Disable AHB1 peripherals clock.

- Parameters
- **Periph:** This parameter can be a combination of the following values: (\*) value not defined in all devices.
    - `LL_AHB1_GRP1_PERIPH_DMA1`
    - `LL_AHB1_GRP1_PERIPH_DMA2 (*)`
    - `LL_AHB1_GRP1_PERIPH_SRAM`
    - `LL_AHB1_GRP1_PERIPH_FLASH`
    - `LL_AHB1_GRP1_PERIPH_FMC (*)`
    - `LL_AHB1_GRP1_PERIPH_CRC`
    - `LL_AHB1_GRP1_PERIPH_GPIOH (*)`
    - `LL_AHB1_GRP1_PERIPH_GPIOA`
    - `LL_AHB1_GRP1_PERIPH_GPIOB`
    - `LL_AHB1_GRP1_PERIPH_GPIOC`
    - `LL_AHB1_GRP1_PERIPH_GPIOD`
    - `LL_AHB1_GRP1_PERIPH_GPIOE (*)`
    - `LL_AHB1_GRP1_PERIPH_GPIOF`
    - `LL_AHB1_GRP1_PERIPH_GPIOG (*)`
    - `LL_AHB1_GRP1_PERIPH_TSC`
    - `LL_AHB1_GRP1_PERIPH_ADC1 (*)`
    - `LL_AHB1_GRP1_PERIPH_ADC12 (*)`
    - `LL_AHB1_GRP1_PERIPH_ADC34 (*)`

- Return values
- **None**

- Reference Manual to  
LL API cross  
reference:
- `AHBENR DMA1EN LL_AHB1_GRP1_DisableClock`
  - `AHBENR DMA2EN LL_AHB1_GRP1_DisableClock`
  - `AHBENR SRAMEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR FLITFEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR FMCEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR CRCEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR GPIOHEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR GPIOAEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR GPIOBEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR GPIOCEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR GPIODEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR GPIOEEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR GPIOFEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR GPIOGEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR TSCEN LL_AHB1_GRP1_DisableClock`
  - `AHBENR ADC1EN LL_AHB1_GRP1_DisableClock`
  - `AHBENR ADC12EN LL_AHB1_GRP1_DisableClock`
  - `AHBENR ADC34EN LL_AHB1_GRP1_DisableClock`

**LL\_AHB1\_GRP1\_ForceReset**

Function name      **`_STATIC_INLINE void LL_AHB1_GRP1_ForceReset (uint32_t  
Periph)`**

Function description      Force AHB1 peripherals reset.

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- LL_AHB1_GRP1_PERIPH_ALL</li> <li>- LL_AHB1_GRP1_PERIPH_FMC (*)</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOH (*)</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOA</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOB</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOC</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOD</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOE (*)</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOF</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOG (*)</li> <li>- LL_AHB1_GRP1_PERIPH_TSC</li> <li>- LL_AHB1_GRP1_PERIPH_ADC1 (*)</li> <li>- LL_AHB1_GRP1_PERIPH_ADC12 (*)</li> <li>- LL_AHB1_GRP1_PERIPH_ADC34 (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• AHBRSTR FMCRST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR GPIOHRST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR GPIOARST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR GPIOBRST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR GPIOCRST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR GPIODRST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR GPIOERST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR GPIOFRST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR GPIOGRST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR TCSRST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR ADC1RST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR ADC12RST LL_AHB1_GRP1_ForceReset</li> <li>• AHBRSTR ADC34RST LL_AHB1_GRP1_ForceReset</li> </ul>                                              |

## LL\_AHB1\_GRP1\_ReleaseReset

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_AHB1_GRP1_ReleaseReset<br/>(uint32_t Periph)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Release AHB1 peripherals reset.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- LL_AHB1_GRP1_PERIPH_ALL</li> <li>- LL_AHB1_GRP1_PERIPH_FMC (*)</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOH (*)</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOA</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOB</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOC</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOD</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOE (*)</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOF</li> <li>- LL_AHB1_GRP1_PERIPH_GPIOG (*)</li> <li>- LL_AHB1_GRP1_PERIPH_TSC</li> <li>- LL_AHB1_GRP1_PERIPH_ADC1 (*)</li> <li>- LL_AHB1_GRP1_PERIPH_ADC12 (*)</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | – LL_AHB1_GRP1_PERIPH_ADC34 (*)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• AHBRSTR FMCRST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR GPIOHRST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR GPIOARST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR GPIOBRST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR GPIOCRST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR GPIODRST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR GPIOERST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR GPIOFRST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR GPIOGRST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR TSCRST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR ADC1RST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR ADC12RST LL_AHB1_GRP1_ReleaseReset</li> <li>• AHBRSTR ADC34RST LL_AHB1_GRP1_ReleaseReset</li> </ul> |

### LL\_APB1\_GRP1\_EnableClock

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_APB1_GRP1_EnableClock<br/>(uint32_t Periph)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Enable APB1 peripherals clock.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>– LL_APB1_GRP1_PERIPH_TIM2</li> <li>– LL_APB1_GRP1_PERIPH_TIM3 (*)</li> <li>– LL_APB1_GRP1_PERIPH_TIM4 (*)</li> <li>– LL_APB1_GRP1_PERIPH_TIM5 (*)</li> <li>– LL_APB1_GRP1_PERIPH_TIM6</li> <li>– LL_APB1_GRP1_PERIPH_TIM7 (*)</li> <li>– LL_APB1_GRP1_PERIPH_TIM12 (*)</li> <li>– LL_APB1_GRP1_PERIPH_TIM13 (*)</li> <li>– LL_APB1_GRP1_PERIPH_TIM14 (*)</li> <li>– LL_APB1_GRP1_PERIPH_TIM18 (*)</li> <li>– LL_APB1_GRP1_PERIPH_WWDG</li> <li>– LL_APB1_GRP1_PERIPH_SPI2 (*)</li> <li>– LL_APB1_GRP1_PERIPH_SPI3 (*)</li> <li>– LL_APB1_GRP1_PERIPH_USART2</li> <li>– LL_APB1_GRP1_PERIPH_USART3</li> <li>– LL_APB1_GRP1_PERIPH_UART4 (*)</li> <li>– LL_APB1_GRP1_PERIPH_UART5 (*)</li> <li>– LL_APB1_GRP1_PERIPH_I2C1</li> <li>– LL_APB1_GRP1_PERIPH_I2C2 (*)</li> <li>– LL_APB1_GRP1_PERIPH_USB (*)</li> <li>– LL_APB1_GRP1_PERIPH_CAN (*)</li> <li>– LL_APB1_GRP1_PERIPH_DAC2 (*)</li> <li>– LL_APB1_GRP1_PERIPH_PWR</li> <li>– LL_APB1_GRP1_PERIPH_DAC1</li> <li>– LL_APB1_GRP1_PERIPH_CEC (*)</li> <li>– LL_APB1_GRP1_PERIPH_I2C3 (*)</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

- Reference Manual to  
LL API cross  
reference:
- APB1ENR TIM2EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR TIM3EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR TIM4EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR TIM5EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR TIM6EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR TIM7EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR TIM12EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR TIM13EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR TIM14EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR TIM18EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR WWDGEN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR SPI2EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR SPI3EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR USART2EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR USART3EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR UART4EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR UART5EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR I2C1EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR I2C2EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR USBEN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR CANEN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR DAC2EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR PWREN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR DAC1EN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR CECEN LL\_APB1\_GRP1\_EnableClock
  - APB1ENR I2C3EN LL\_APB1\_GRP1\_EnableClock

### **LL\_APB1\_GRP1\_IsEnabledClock**

Function name      **\_STATIC\_INLINE uint32\_t LL\_APB1\_GRP1\_IsEnabledClock  
(uint32\_t Periph)**

Function description      Check if APB1 peripheral clock is enabled or not.

- Parameters
- **Periph:** This parameter can be a combination of the following values: (\*) value not defined in all devices.
    - LL\_APB1\_GRP1\_PERIPH\_TIM2
    - LL\_APB1\_GRP1\_PERIPH\_TIM3 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_TIM4 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_TIM5 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_TIM6
    - LL\_APB1\_GRP1\_PERIPH\_TIM7 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_TIM12 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_TIM13 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_TIM14 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_TIM18 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_WWDG
    - LL\_APB1\_GRP1\_PERIPH\_SPI2 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_SPI3 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_USART2
    - LL\_APB1\_GRP1\_PERIPH\_USART3
    - LL\_APB1\_GRP1\_PERIPH\_UART4 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_UART5 (\*)
    - LL\_APB1\_GRP1\_PERIPH\_I2C1

- LL\_APB1\_GRP1\_PERIPH\_I2C2 (\*)
- LL\_APB1\_GRP1\_PERIPH\_USB (\*)
- LL\_APB1\_GRP1\_PERIPH\_CAN (\*)
- LL\_APB1\_GRP1\_PERIPH\_DAC2 (\*)
- LL\_APB1\_GRP1\_PERIPH\_PWR
- LL\_APB1\_GRP1\_PERIPH\_DAC1
- LL\_APB1\_GRP1\_PERIPH\_CEC (\*)
- LL\_APB1\_GRP1\_PERIPH\_I2C3 (\*)

**Return values**

- **State:** of Periph (1 or 0).

**Reference Manual to  
LL API cross  
reference:**

- APB1ENR TIM2EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR TIM3EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR TIM4EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR TIM5EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR TIM6EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR TIM7EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR TIM12EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR TIM13EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR TIM14EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR TIM18EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR WWDGEN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR SPI2EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR SPI3EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR USART2EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR USART3EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR UART4EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR UART5EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR I2C1EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR I2C2EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR USBEN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR CANEN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR DAC2EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR PWREN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR DAC1EN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR CECEN LL\_APB1\_GRP1\_IsEnabledClock
- APB1ENR I2C3EN LL\_APB1\_GRP1\_IsEnabledClock

**LL\_APB1\_GRP1\_DisableClock****Function name**

**\_STATIC\_INLINE void LL\_APB1\_GRP1\_DisableClock  
(uint32\_t Periph)**

**Function description**

Disable APB1 peripherals clock.

**Parameters**

- **Periph:** This parameter can be a combination of the following values: (\*) value not defined in all devices.
  - LL\_APB1\_GRP1\_PERIPH\_TIM2
  - LL\_APB1\_GRP1\_PERIPH\_TIM3 (\*)
  - LL\_APB1\_GRP1\_PERIPH\_TIM4 (\*)
  - LL\_APB1\_GRP1\_PERIPH\_TIM5 (\*)
  - LL\_APB1\_GRP1\_PERIPH\_TIM6
  - LL\_APB1\_GRP1\_PERIPH\_TIM7 (\*)
  - LL\_APB1\_GRP1\_PERIPH\_TIM12 (\*)
  - LL\_APB1\_GRP1\_PERIPH\_TIM13 (\*)

- LL\_APB1\_GRP1\_PERIPH\_TIM14 (\*)
- LL\_APB1\_GRP1\_PERIPH\_TIM18 (\*)
- LL\_APB1\_GRP1\_PERIPH\_WWDG
- LL\_APB1\_GRP1\_PERIPH\_SPI2 (\*)
- LL\_APB1\_GRP1\_PERIPH\_SPI3 (\*)
- LL\_APB1\_GRP1\_PERIPH\_USART2
- LL\_APB1\_GRP1\_PERIPH\_USART3
- LL\_APB1\_GRP1\_PERIPH\_UART4 (\*)
- LL\_APB1\_GRP1\_PERIPH\_UART5 (\*)
- LL\_APB1\_GRP1\_PERIPH\_I2C1
- LL\_APB1\_GRP1\_PERIPH\_I2C2 (\*)
- LL\_APB1\_GRP1\_PERIPH\_USB (\*)
- LL\_APB1\_GRP1\_PERIPH\_CAN (\*)
- LL\_APB1\_GRP1\_PERIPH\_DAC2 (\*)
- LL\_APB1\_GRP1\_PERIPH\_PWR
- LL\_APB1\_GRP1\_PERIPH\_DAC1
- LL\_APB1\_GRP1\_PERIPH\_CEC (\*)
- LL\_APB1\_GRP1\_PERIPH\_I2C3 (\*)

**Return values**

Reference Manual to  
LL API cross  
reference:

- **None**

- APB1ENR TIM2EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR TIM3EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR TIM4EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR TIM5EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR TIM6EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR TIM7EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR TIM12EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR TIM13EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR TIM14EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR TIM18EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR WWDGEN LL\_APB1\_GRP1\_DisableClock
- APB1ENR SPI2EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR SPI3EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR USART2EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR USART3EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR UART4EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR UART5EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR I2C1EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR I2C2EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR USBEN LL\_APB1\_GRP1\_DisableClock
- APB1ENR CANEN LL\_APB1\_GRP1\_DisableClock
- APB1ENR DAC2EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR PWREN LL\_APB1\_GRP1\_DisableClock
- APB1ENR DAC1EN LL\_APB1\_GRP1\_DisableClock
- APB1ENR CECEN LL\_APB1\_GRP1\_DisableClock
- APB1ENR I2C3EN LL\_APB1\_GRP1\_DisableClock

**LL\_APB1\_GRP1\_ForceReset**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_APB1_GRP1_ForceReset (uint32_t Periph)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description                              | Force APB1 peripherals reset.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- <code>LL_APB1_GRP1_PERIPH_ALL</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_TIM2</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_TIM3 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_TIM4 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_TIM5 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_TIM6</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_TIM7 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_TIM12 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_TIM13 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_TIM14 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_TIM18 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_WWDG</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_SPI2 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_SPI3 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_USART2</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_USART3</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_UART4 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_UART5 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_I2C1</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_I2C2 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_USB (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_CAN (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_DAC2 (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_PWR</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_DAC1</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_CEC (*)</code></li> <li>- <code>LL_APB1_GRP1_PERIPH_I2C3 (*)</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>APB1RSTR_TIM2RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_TIM3RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_TIM4RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_TIM5RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_TIM6RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_TIM7RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_TIM12RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_TIM13RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_TIM14RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_TIM18RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_WWDGRST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_SPI2RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_SPI3RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_USART2RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_USART3RST LL_APB1_GRP1_ForceReset</code></li> <li>• <code>APB1RSTR_UART4RST LL_APB1_GRP1_ForceReset</code></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

- APB1RSTR UART5RST LL\_APB1\_GRP1\_ForceReset
- APB1RSTR I2C1RST LL\_APB1\_GRP1\_ForceReset
- APB1RSTR I2C2RST LL\_APB1\_GRP1\_ForceReset
- APB1RSTR USBRST LL\_APB1\_GRP1\_ForceReset
- APB1RSTR CANRST LL\_APB1\_GRP1\_ForceReset
- APB1RSTR DAC2RST LL\_APB1\_GRP1\_ForceReset
- APB1RSTR PWRRST LL\_APB1\_GRP1\_ForceReset
- APB1RSTR DAC1RST LL\_APB1\_GRP1\_ForceReset
- APB1RSTR CECRST LL\_APB1\_GRP1\_ForceReset
- APB1RSTR I2C3RST LL\_APB1\_GRP1\_ForceReset

### LL\_APB1\_GRP1\_ReleaseReset

Function name **`__STATIC_INLINE void LL_APB1_GRP1_ReleaseReset(uint32_t Periph)`**

Function description Release APB1 peripherals reset.

Parameters
 

- **Periph:** This parameter can be a combination of the following values: (\*) value not defined in all devices.

- LL\_APB1\_GRP1\_PERIPH\_ALL
- LL\_APB1\_GRP1\_PERIPH\_TIM2
- LL\_APB1\_GRP1\_PERIPH\_TIM3 (\*)
- LL\_APB1\_GRP1\_PERIPH\_TIM4 (\*)
- LL\_APB1\_GRP1\_PERIPH\_TIM5 (\*)
- LL\_APB1\_GRP1\_PERIPH\_TIM6
- LL\_APB1\_GRP1\_PERIPH\_TIM7 (\*)
- LL\_APB1\_GRP1\_PERIPH\_TIM12 (\*)
- LL\_APB1\_GRP1\_PERIPH\_TIM13 (\*)
- LL\_APB1\_GRP1\_PERIPH\_TIM14 (\*)
- LL\_APB1\_GRP1\_PERIPH\_TIM18 (\*)
- LL\_APB1\_GRP1\_PERIPH\_WWDG
- LL\_APB1\_GRP1\_PERIPH\_SPI2 (\*)
- LL\_APB1\_GRP1\_PERIPH\_SPI3 (\*)
- LL\_APB1\_GRP1\_PERIPH\_USART2
- LL\_APB1\_GRP1\_PERIPH\_USART3
- LL\_APB1\_GRP1\_PERIPH\_UART4 (\*)
- LL\_APB1\_GRP1\_PERIPH\_UART5 (\*)
- LL\_APB1\_GRP1\_PERIPH\_I2C1
- LL\_APB1\_GRP1\_PERIPH\_I2C2 (\*)
- LL\_APB1\_GRP1\_PERIPH\_USB (\*)
- LL\_APB1\_GRP1\_PERIPH\_CAN (\*)
- LL\_APB1\_GRP1\_PERIPH\_DAC2 (\*)
- LL\_APB1\_GRP1\_PERIPH\_PWR
- LL\_APB1\_GRP1\_PERIPH\_DAC1
- LL\_APB1\_GRP1\_PERIPH\_CEC (\*)
- LL\_APB1\_GRP1\_PERIPH\_I2C3 (\*)

Return values
 

- **None**

Reference Manual to  
LL API cross  
reference:
 

- APB1RSTR TIM2RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR TIM3RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR TIM4RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR TIM5RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR TIM6RST LL\_APB1\_GRP1\_ReleaseReset

- APB1RSTR TIM7RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR TIM12RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR TIM13RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR TIM14RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR TIM18RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR WWDGRST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR SPI2RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR SPI3RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR USART2RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR USART3RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR UART4RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR UART5RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR I2C1RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR I2C2RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR USBRST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR CANRST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR DAC2RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR PWRRST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR DAC1RST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR CECRST LL\_APB1\_GRP1\_ReleaseReset
- APB1RSTR I2C3RST LL\_APB1\_GRP1\_ReleaseReset

### **LL\_APB2\_GRP1\_EnableClock**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>STATIC_INLINE void LL_APB2_GRP1_EnableClock<br/>(uint32_t Periph)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Enable APB2 peripherals clock.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- LL_APB2_GRP1_PERIPH_SYSCFG</li> <li>- LL_APB2_GRP1_PERIPH_ADC1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SPI1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM8 (*)</li> <li>- LL_APB2_GRP1_PERIPH_USART1</li> <li>- LL_APB2_GRP1_PERIPH_SPI4 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM15</li> <li>- LL_APB2_GRP1_PERIPH_TIM16</li> <li>- LL_APB2_GRP1_PERIPH_TIM17</li> <li>- LL_APB2_GRP1_PERIPH_TIM19 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM20 (*)</li> <li>- LL_APB2_GRP1_PERIPH_HRTIM1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SDADC1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SDADC2 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SDADC3 (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• APB2ENR SYSCFGEN LL_APB2_GRP1_EnableClock</li> <li>• APB2ENR ADC1EN LL_APB2_GRP1_EnableClock</li> <li>• APB2ENR TIM1EN LL_APB2_GRP1_EnableClock</li> <li>• APB2ENR SPI1EN LL_APB2_GRP1_EnableClock</li> <li>• APB2ENR TIM8EN LL_APB2_GRP1_EnableClock</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

- APB2ENR USART1EN LL\_APB2\_GRP1\_EnableClock
- APB2ENR SPI4EN LL\_APB2\_GRP1\_EnableClock
- APB2ENR TIM15EN LL\_APB2\_GRP1\_EnableClock
- APB2ENR TIM16EN LL\_APB2\_GRP1\_EnableClock
- APB2ENR TIM17EN LL\_APB2\_GRP1\_EnableClock
- APB2ENR TIM19EN LL\_APB2\_GRP1\_EnableClock
- APB2ENR TIM20EN LL\_APB2\_GRP1\_EnableClock
- APB2ENR HRTIM1EN LL\_APB2\_GRP1\_EnableClock
- APB2ENR SDADC1EN LL\_APB2\_GRP1\_EnableClock
- APB2ENR SDADC2EN LL\_APB2\_GRP1\_EnableClock
- APB2ENR SDADC3EN LL\_APB2\_GRP1\_EnableClock

### **LL\_APB2\_GRP1\_IsEnabledClock**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_APB2_GRP1_IsEnabledClock(uint32_t Periph)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Check if APB2 peripheral clock is enabled or not.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- LL_APB2_GRP1_PERIPH_SYSCFG</li> <li>- LL_APB2_GRP1_PERIPH_ADC1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SPI1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM8 (*)</li> <li>- LL_APB2_GRP1_PERIPH_USART1</li> <li>- LL_APB2_GRP1_PERIPH_SPI4 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM15</li> <li>- LL_APB2_GRP1_PERIPH_TIM16</li> <li>- LL_APB2_GRP1_PERIPH_TIM17</li> <li>- LL_APB2_GRP1_PERIPH_TIM19 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM20 (*)</li> <li>- LL_APB2_GRP1_PERIPH_HRTIM1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SDADC1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SDADC2 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SDADC3 (*)</li> </ul> </li> </ul>                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of Periph (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• APB2ENR SYSCFGEN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR ADC1EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR TIM1EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR SPI1EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR TIM8EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR USART1EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR SPI4EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR TIM15EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR TIM16EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR TIM17EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR TIM19EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR TIM20EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR HRTIM1EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR SDADC1EN LL_APB2_GRP1_IsEnabledClock</li> <li>• APB2ENR SDADC2EN LL_APB2_GRP1_IsEnabledClock</li> </ul> |

- APB2ENR SDADC3EN LL\_APB2\_GRP1\_IsEnabledClock

### LL\_APB2\_GRP1\_DisableClock

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_APB2_GRP1_DisableClock (uint32_t Periph)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description                              | Disable APB2 peripherals clock.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- LL_APB2_GRP1_PERIPH_SYSCFG</li> <li>- LL_APB2_GRP1_PERIPH_ADC1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SPI1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM8 (*)</li> <li>- LL_APB2_GRP1_PERIPH_USART1</li> <li>- LL_APB2_GRP1_PERIPH_SPI4 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM15</li> <li>- LL_APB2_GRP1_PERIPH_TIM16</li> <li>- LL_APB2_GRP1_PERIPH_TIM17</li> <li>- LL_APB2_GRP1_PERIPH_TIM19 (*)</li> <li>- LL_APB2_GRP1_PERIPH_TIM20 (*)</li> <li>- LL_APB2_GRP1_PERIPH_HRTIM1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SDADC1 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SDADC2 (*)</li> <li>- LL_APB2_GRP1_PERIPH_SDADC3 (*)</li> </ul> </li> </ul>                                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• APB2ENR SYSCFGEN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR ADC1EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR TIM1EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR SPI1EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR TIM8EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR USART1EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR SPI4EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR TIM15EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR TIM16EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR TIM17EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR TIM19EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR TIM20EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR HRTIM1EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR SDADC1EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR SDADC2EN LL_APB2_GRP1_DisableClock</li> <li>• APB2ENR SDADC3EN LL_APB2_GRP1_DisableClock</li> </ul> |

### LL\_APB2\_GRP1\_ForceReset

|                      |                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_APB2_GRP1_ForceReset (uint32_t Periph)</code>                                                                                           |
| Function description | Force APB2 peripherals reset.                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices.</li> </ul> |

- LL\_APB2\_GRP1\_PERIPH\_ALL
- LL\_APB2\_GRP1\_PERIPH\_SYSCFG
- LL\_APB2\_GRP1\_PERIPH\_ADC1 (\*)
- LL\_APB2\_GRP1\_PERIPH\_TIM1 (\*)
- LL\_APB2\_GRP1\_PERIPH\_SPI1 (\*)
- LL\_APB2\_GRP1\_PERIPH\_TIM8 (\*)
- LL\_APB2\_GRP1\_PERIPH\_USART1
- LL\_APB2\_GRP1\_PERIPH\_SPI4 (\*)
- LL\_APB2\_GRP1\_PERIPH\_TIM15
- LL\_APB2\_GRP1\_PERIPH\_TIM16
- LL\_APB2\_GRP1\_PERIPH\_TIM17
- LL\_APB2\_GRP1\_PERIPH\_TIM19 (\*)
- LL\_APB2\_GRP1\_PERIPH\_TIM20 (\*)
- LL\_APB2\_GRP1\_PERIPH\_HRTIM1 (\*)
- LL\_APB2\_GRP1\_PERIPH\_SDADC1 (\*)
- LL\_APB2\_GRP1\_PERIPH\_SDADC2 (\*)
- LL\_APB2\_GRP1\_PERIPH\_SDADC3 (\*)

**Return values**

Reference Manual to  
LL API cross  
reference:

- **None**

- APB2RSTR SYSCFGRST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR ADC1RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR TIM1RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR SPI1RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR TIM8RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR USART1RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR SPI4RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR TIM15RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR TIM16RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR TIM17RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR TIM19RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR TIM20RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR HRTIM1RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR SDADC1RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR SDADC2RST LL\_APB2\_GRP1\_ForceReset
- APB2RSTR SDADC3RST LL\_APB2\_GRP1\_ForceReset

### LL\_APB2\_GRP1\_ReleaseReset

Function name **STATIC\_INLINE void LL\_APB2\_GRP1\_ReleaseReset (uint32\_t Periph)**

Function description Release APB2 peripherals reset.

- Parameters
- **Periph:** This parameter can be a combination of the following values: (\*) value not defined in all devices.
    - LL\_APB2\_GRP1\_PERIPH\_ALL
    - LL\_APB2\_GRP1\_PERIPH\_SYSCFG
    - LL\_APB2\_GRP1\_PERIPH\_ADC1 (\*)
    - LL\_APB2\_GRP1\_PERIPH\_TIM1 (\*)
    - LL\_APB2\_GRP1\_PERIPH\_SPI1 (\*)
    - LL\_APB2\_GRP1\_PERIPH\_TIM8 (\*)
    - LL\_APB2\_GRP1\_PERIPH\_USART1
    - LL\_APB2\_GRP1\_PERIPH\_SPI4 (\*)
    - LL\_APB2\_GRP1\_PERIPH\_TIM15

- LL\_APB2\_GRP1\_PERIPH\_TIM16
- LL\_APB2\_GRP1\_PERIPH\_TIM17
- LL\_APB2\_GRP1\_PERIPH\_TIM19 (\*)
- LL\_APB2\_GRP1\_PERIPH\_TIM20 (\*)
- LL\_APB2\_GRP1\_PERIPH\_HRTIM1 (\*)
- LL\_APB2\_GRP1\_PERIPH\_SDADC1 (\*)
- LL\_APB2\_GRP1\_PERIPH\_SDADC2 (\*)
- LL\_APB2\_GRP1\_PERIPH\_SDADC3 (\*)

**Return values**

- **None**

Reference Manual to  
LL API cross  
reference:

- APB2RSTR SYSCFGRST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR ADC1RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR TIM1RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR SPI1RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR TIM8RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR USART1RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR SPI4RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR TIM15RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR TIM16RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR TIM17RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR TIM19RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR TIM20RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR HRTIM1RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR SDADC1RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR SDADC2RST LL\_APB2\_GRP1\_ReleaseReset
- APB2RSTR SDADC3RST LL\_APB2\_GRP1\_ReleaseReset

## 60.2 BUS Firmware driver defines

### 60.2.1 BUS

#### AHB1 GRP1 PERIPH

```
LL_AHB1_GRP1_PERIPH_ALL
LL_AHB1_GRP1_PERIPH_DMA1
LL_AHB1_GRP1_PERIPH_DMA2
LL_AHB1_GRP1_PERIPH_SRAM
LL_AHB1_GRP1_PERIPH_FLASH
LL_AHB1_GRP1_PERIPH_CRC
LL_AHB1_GRP1_PERIPH_GPIOA
LL_AHB1_GRP1_PERIPH_GPIOB
LL_AHB1_GRP1_PERIPH_GPIOC
LL_AHB1_GRP1_PERIPH_GPIOD
LL_AHB1_GRP1_PERIPH_GPIOE
LL_AHB1_GRP1_PERIPH_GPIOF
LL_AHB1_GRP1_PERIPH_TSC
LL_AHB1_GRP1_PERIPH_ADC12
```

LL\_AHB1\_GRP1\_PERIPH\_ADC34

**APB1 GRP1 PERIPH**

LL\_APB1\_GRP1\_PERIPH\_ALL

LL\_APB1\_GRP1\_PERIPH\_TIM2

LL\_APB1\_GRP1\_PERIPH\_TIM3

LL\_APB1\_GRP1\_PERIPH\_TIM4

LL\_APB1\_GRP1\_PERIPH\_TIM6

LL\_APB1\_GRP1\_PERIPH\_TIM7

LL\_APB1\_GRP1\_PERIPH\_WWDG

LL\_APB1\_GRP1\_PERIPH\_SPI2

LL\_APB1\_GRP1\_PERIPH\_SPI3

LL\_APB1\_GRP1\_PERIPH\_USART2

LL\_APB1\_GRP1\_PERIPH\_USART3

LL\_APB1\_GRP1\_PERIPH\_UART4

LL\_APB1\_GRP1\_PERIPH\_UART5

LL\_APB1\_GRP1\_PERIPH\_I2C1

LL\_APB1\_GRP1\_PERIPH\_I2C2

LL\_APB1\_GRP1\_PERIPH\_USB

LL\_APB1\_GRP1\_PERIPH\_CAN

LL\_APB1\_GRP1\_PERIPH\_PWR

LL\_APB1\_GRP1\_PERIPH\_DAC1

**APB2 GRP1 PERIPH**

LL\_APB2\_GRP1\_PERIPH\_ALL

LL\_APB2\_GRP1\_PERIPH\_SYSCFG

LL\_APB2\_GRP1\_PERIPH\_TIM1

LL\_APB2\_GRP1\_PERIPH\_SPI1

LL\_APB2\_GRP1\_PERIPH\_TIM8

LL\_APB2\_GRP1\_PERIPH\_USART1

LL\_APB2\_GRP1\_PERIPH\_TIM15

LL\_APB2\_GRP1\_PERIPH\_TIM16

LL\_APB2\_GRP1\_PERIPH\_TIM17

## 61 LL COMP Generic Driver

### 61.1 COMP Firmware driver registers structures

#### 61.1.1 LL\_COMP\_InitTypeDef

##### Data Fields

- *uint32\_t PowerMode*
- *uint32\_t InputPlus*
- *uint32\_t InputMinus*
- *uint32\_t InputHysteresis*
- *uint32\_t OutputSelection*
- *uint32\_t OutputPolarity*
- *uint32\_t OutputBlankingSource*

##### Field Documentation

- ***uint32\_t LL\_COMP\_InitTypeDef::PowerMode***  
Set comparator operating mode to adjust power and speed. This parameter can be a value of **COMP\_LL\_EC\_POWERMODE**This feature can be modified afterwards using unitary function **LL\_COMP\_SetPowerMode()**.
- ***uint32\_t LL\_COMP\_InitTypeDef::InputPlus***  
Set comparator input plus (non-inverting input). This parameter can be a value of **COMP\_LL\_EC\_INPUT\_PLUS**This feature can be modified afterwards using unitary function **LL\_COMP\_SetInputPlus()**.
- ***uint32\_t LL\_COMP\_InitTypeDef::InputMinus***  
Set comparator input minus (inverting input). This parameter can be a value of **COMP\_LL\_EC\_INPUT\_MINUS**This feature can be modified afterwards using unitary function **LL\_COMP\_SetInputMinus()**.
- ***uint32\_t LL\_COMP\_InitTypeDef::InputHysteresis***  
Set comparator hysteresis mode of the input minus. This parameter can be a value of **COMP\_LL\_EC\_INPUT\_HYSTERESIS**This feature can be modified afterwards using unitary function **LL\_COMP\_SetInputHysteresis()**.
- ***uint32\_t LL\_COMP\_InitTypeDef::OutputSelection***  
Set comparator output selection. This parameter can be a value of **COMP\_LL\_EC\_OUTPUT\_SELECTION**This feature can be modified afterwards using unitary function **LL\_COMP\_SetOutputSelection()**.
- ***uint32\_t LL\_COMP\_InitTypeDef::OutputPolarity***  
Set comparator output polarity. This parameter can be a value of **COMP\_LL\_EC\_OUTPUT\_POLARITY**This feature can be modified afterwards using unitary function **LL\_COMP\_SetOutputPolarity()**.
- ***uint32\_t LL\_COMP\_InitTypeDef::OutputBlankingSource***  
Set comparator blanking source. This parameter can be a value of **COMP\_LL\_EC\_OUTPUT\_BLANKING\_SOURCE**This feature can be modified afterwards using unitary function **LL\_COMP\_SetOutputBlankingSource()**.

## 61.2 COMP Firmware driver API description

### 61.2.1 Detailed description of functions

#### LL\_COMP\_SetCommonWindowMode

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE void LL_COMP_SetCommonWindowMode<br/>(COMP_Common_TypeDef * COMPxy_COMMON, uint32_t<br/>WindowMode)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                        | Set window mode of a pair of comparators instances (2 consecutive COMP instances odd and even COMP<x> and COMP<x+1>).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>COMPxy_COMMON:</b> Comparator common instance (can be set directly from CMSIS definition or by using helper macro <code>_LL_COMP_COMMON_INSTANCE()</code>)</li> <li><b>WindowMode:</b> This parameter can be one of the following values: (1) Parameter available on devices: STM32F302xB/C, STM32F303xB/C, STM32F358xC (2) Parameter available on devices: STM32F303xB/C, STM32F358xC <ul style="list-style-type: none"> <li>- <code>LL_COMP_WINDOWMODE_DISABLE</code></li> <li>- <code>LL_COMP_WINDOWMODE_COMP1_INPUT_PLUS_COMMON</code> (1)</li> <li>- <code>LL_COMP_WINDOWMODE_COMP3_INPUT_PLUS_COMMON</code> (2)</li> <li>- <code>LL_COMP_WINDOWMODE_COMP5_INPUT_PLUS_COMMON</code> (2)</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CSR COMPxWNDWEN LL_COMP_SetCommonWindowMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

#### LL\_COMP\_GetCommonWindowMode

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_COMP_GetCommonWindowMode<br/>(COMP_Common_TypeDef * COMPxy_COMMON)</code>                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Get window mode of a pair of comparators instances (2 consecutive COMP instances odd and even COMP<x> and COMP<x+1>).                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li><b>COMPxy_COMMON:</b> Comparator common instance (can be set directly from CMSIS definition or by using helper macro <code>_LL_COMP_COMMON_INSTANCE()</code>)</li> </ul>                                                                                                                                                                                                                                                                                      |
| Return values        | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values: (1) Parameter available on devices: STM32F302xB/C, STM32F303xB/C, STM32F358xC (2) Parameter available on devices: STM32F303xB/C, STM32F358xC <ul style="list-style-type: none"> <li>- <code>LL_COMP_WINDOWMODE_DISABLE</code></li> <li>- <code>LL_COMP_WINDOWMODE_COMP1_INPUT_PLUS_COMMON</code> (1)</li> <li>- <code>LL_COMP_WINDOWMODE_COMP3_INPUT_PLUS_COMMON</code> (2)</li> </ul> </li> </ul> |

- LL\_COMP\_WINDOWMODE\_COMP5\_INPUT\_PLUS\_COMMON  
(2)
  - CSR COMPxWNDWEN LL\_COMP\_GetCommonWindowMode
- Reference Manual to LL API cross reference:

### **LL\_COMP\_SetPowerMode**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>STATIC_INLINE void LL_COMP_SetPowerMode (COMP_TypeDef * COMPx, uint32_t PowerMode)</code></b>                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                        | Set comparator instance operating mode to adjust power and speed.                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> <li>• <b>PowerMode:</b> This parameter can be one of the following values: (1) Parameter available only on devices: STM32F302xB/C, STM32F303xB/C, STM32F358xC <ul style="list-style-type: none"> <li>- LL_COMP_POWERMODE_HIGHSPEED</li> <li>- LL_COMP_POWERMODE_MEDIUMSPEED (1)</li> <li>- LL_COMP_POWERMODE_LOWPOWER (1)</li> <li>- LL_COMP_POWERMODE_ULTRALOWPOWER (1)</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR COMPxMODE LL_COMP_SetPowerMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                          |

### **LL\_COMP\_GetPowerMode**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>STATIC_INLINE uint32_t LL_COMP_GetPowerMode (COMP_TypeDef * COMPx)</code></b>                                                                                                                                                                                                                                                                                                                                   |
| Function description                        | Get comparator instance operating mode to adjust power and speed.                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul>                                                                                                                                                                                                                                                                                                                                    |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (1) Parameter available only on devices: STM32F302xB/C, STM32F303xB/C, STM32F358xC <ul style="list-style-type: none"> <li>- LL_COMP_POWERMODE_HIGHSPEED</li> <li>- LL_COMP_POWERMODE_MEDIUMSPEED (1)</li> <li>- LL_COMP_POWERMODE_LOWPOWER (1)</li> <li>- LL_COMP_POWERMODE_ULTRALOWPOWER (1)</li> </ul> </li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR COMPxMODE LL_COMP_GetPowerMode</li> </ul>                                                                                                                                                                                                                                                                                                                                   |

**LL\_COMP\_ConfigInputs**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_COMP_ConfigInputs<br/>(COMP_TypeDef * COMPx, uint32_t InputMinus, uint32_t<br/>InputPlus)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Set comparator inputs minus (inverting) and plus (non-inverting).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> <li>• <b>InputMinus:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_COMP_INPUT_MINUS_1_4VREFINT</li> <li>– LL_COMP_INPUT_MINUS_1_2VREFINT</li> <li>– LL_COMP_INPUT_MINUS_3_4VREFINT</li> <li>– LL_COMP_INPUT_MINUS_VREFINT</li> <li>– LL_COMP_INPUT_MINUS_DAC1_CH1</li> <li>– LL_COMP_INPUT_MINUS_DAC1_CH2 (3)</li> <li>– LL_COMP_INPUT_MINUS_DAC2_CH1 (2)</li> <li>– LL_COMP_INPUT_MINUS_IO1</li> <li>– LL_COMP_INPUT_MINUS_IO2</li> <li>– LL_COMP_INPUT_MINUS_IO3 (1)</li> <li>– LL_COMP_INPUT_MINUS_IO4 (1) Parameter available on all devices except STM32F301x6/8, STM32F318x8, STM32F302x6/8, STM32F303x6/8, STM32F328xx, STM32F334xx.</li> <li>– (2) Parameter available only on devices STM32F303x6/8, STM32F328x8, STM32F334xx.</li> <li>– (3) Parameter available on all devices except STM32F301x6/8, STM32F318x8, STM32F302xx.</li> <li>–</li> </ul> </li> <li>• <b>InputPlus:</b> This parameter can be one of the following values: (1) Parameter available only on devices STM32F302xB/C, STM32F303xB/C, STM32F358xC. <ul style="list-style-type: none"> <li>– LL_COMP_INPUT_PLUS_IO1</li> <li>– LL_COMP_INPUT_PLUS_IO2 (1)</li> <li>– LL_COMP_INPUT_PLUS_DAC1_CH1_COMP1 (2)</li> <li>– LL_COMP_INPUT_PLUS_DAC1_CH1_COMP2 (3)</li> </ul> </li> <li>• (2) Parameter available on devices: STM32F302xB/C, STM32F302xD/E, STM32F303xB/C/D/E, STM32F358xC, STM32F398xE.</li> <li>• (3) Parameter available on devices: STM32F301x6/8, STM32F318xx, STM32F302x6/8.</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• In case of comparator input selected to be connected to IO: GPIO pins are specific to each comparator instance. Refer to description of parameters or to reference manual.</li> <li>• On this STM32 serie, a voltage scaler is used when COMP input is based on VrefInt (VrefInt or subdivision of VrefInt): Voltage scaler requires a delay for voltage stabilization. Refer to device datasheet, parameter "tS_SC".</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR INMSEL LL_COMP_ConfigInputs</li> <li>• CSR NONINSEL LL_COMP_ConfigInputs</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**LL\_COMP\_SetInputPlus**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_COMP_SetInputPlus<br/>(COMP_TypeDef * COMPx, uint32_t InputPlus)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Set comparator input plus (non-inverting).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> <li>• <b>InputPlus:</b> This parameter can be one of the following values: (1) Parameter available only on devices STM32F302xB/C, STM32F303xB/C, STM32F358xC. <ul style="list-style-type: none"> <li>– LL_COMP_INPUT_PLUS_IO1</li> <li>– LL_COMP_INPUT_PLUS_IO2 (1)</li> <li>– LL_COMP_INPUT_PLUS_DAC1_CH1_COMP1 (2)</li> <li>– LL_COMP_INPUT_PLUS_DAC1_CH1_COMP2 (3)</li> </ul> </li> <li>• (2) Parameter available on devices: STM32F302xB/C, STM32F302xD/E, STM32F303xB/C/D/E, STM32F358xC, STM32F398xE.</li> <li>• (3) Parameter available on devices: STM32F301x6/8, STM32F318xx, STM32F302x6/8.</li> </ul> |
| Return values                                     | • <b>None</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• In case of comparator input selected to be connected to IO: GPIO pins are specific to each comparator instance. Refer to description of parameters or to reference manual.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR NONINSEL LL_COMP_SetInputPlus</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**LL\_COMP\_GetInputPlus**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_COMP_GetInputPlus<br/>(COMP_TypeDef * COMPx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Get comparator input plus (non-inverting).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (1) Parameter available only on devices STM32F302xB/C, STM32F303xB/C, STM32F358xC. <ul style="list-style-type: none"> <li>– LL_COMP_INPUT_PLUS_IO1</li> <li>– LL_COMP_INPUT_PLUS_IO2 (1)</li> <li>– LL_COMP_INPUT_PLUS_DAC1_CH1_COMP1 (2)</li> <li>– LL_COMP_INPUT_PLUS_DAC1_CH1_COMP2 (3)</li> </ul> </li> <li>• (2) Parameter available on devices: STM32F302xB/C, STM32F302xD/E, STM32F303xB/C/D/E, STM32F358xC, STM32F398xE.</li> <li>• (3) Parameter available on devices: STM32F301x6/8, STM32F318xx, STM32F302x6/8.</li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• In case of comparator input selected to be connected to IO: GPIO pins are specific to each comparator instance. Refer to description of parameters or to reference manual.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                            |

Reference Manual to  
LL API cross  
reference:

- CSR NONINSEL LL\_COMP\_GetInputPlus

## LL\_COMP\_SetInputMinus

Function name

**\_STATIC\_INLINE void LL\_COMP\_SetInputMinus  
(COMP\_TypeDef \* COMPx, uint32\_t InputMinus)**

Function description

Set comparator input minus (inverting).

Parameters

- COMPx:** Comparator instance
- InputMinus:** This parameter can be one of the following values:
  - LL\_COMP\_INPUT\_MINUS\_1\_4VREFINT
  - LL\_COMP\_INPUT\_MINUS\_1\_2VREFINT
  - LL\_COMP\_INPUT\_MINUS\_3\_4VREFINT
  - LL\_COMP\_INPUT\_MINUS\_VREFINT
  - LL\_COMP\_INPUT\_MINUS\_DAC1\_CH1
  - LL\_COMP\_INPUT\_MINUS\_DAC1\_CH2 (3)
  - LL\_COMP\_INPUT\_MINUS\_DAC2\_CH1 (2)
  - LL\_COMP\_INPUT\_MINUS\_IO1
  - LL\_COMP\_INPUT\_MINUS\_IO2
  - LL\_COMP\_INPUT\_MINUS\_IO3 (1)
  - LL\_COMP\_INPUT\_MINUS\_IO4 (1) Parameter available on all devices except STM32F301x6/8, STM32F318x8, STM32F302x6/8, STM32F303x6/8, STM32F328xx, STM32F334xx.
  - (2) Parameter available only on devices STM32F303x6/8, STM32F328x8, STM32F334xx.
  - (3) Parameter available on all devices except STM32F301x6/8, STM32F318x8, STM32F302xx.
  -

Return values

- None**

Notes

- In case of comparator input selected to be connected to IO: GPIO pins are specific to each comparator instance. Refer to description of parameters or to reference manual.
- On this STM32 serie, a voltage scaler is used when COMP input is based on VrefInt (VrefInt or subdivision of VrefInt): Voltage scaler requires a delay for voltage stabilization. Refer to device datasheet, parameter "tS\_SC".

Reference Manual to  
LL API cross  
reference:

- CSR INMSEL LL\_COMP\_SetInputMinus

**LL\_COMP\_GetInputMinus**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_COMP_GetInputMinus<br/>(COMP_TypeDef * COMPx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description                              | Get comparator input minus (inverting).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_COMP_INPUT_MINUS_1_4VREFINT</li> <li>- LL_COMP_INPUT_MINUS_1_2VREFINT</li> <li>- LL_COMP_INPUT_MINUS_3_4VREFINT</li> <li>- LL_COMP_INPUT_MINUS_VREFINT</li> <li>- LL_COMP_INPUT_MINUS_DAC1_CH1</li> <li>- LL_COMP_INPUT_MINUS_DAC1_CH2 (3)</li> <li>- LL_COMP_INPUT_MINUS_DAC2_CH1 (2)</li> <li>- LL_COMP_INPUT_MINUS_IO1</li> <li>- LL_COMP_INPUT_MINUS_IO2</li> <li>- LL_COMP_INPUT_MINUS_IO3 (1)</li> <li>- LL_COMP_INPUT_MINUS_IO4 (1) Parameter available on all devices except STM32F301x6/8, STM32F318x8, STM32F302x6/8, STM32F303x6/8, STM32F328xx, STM32F334xx.</li> <li>- (2) Parameter available only on devices STM32F303x6/8, STM32F328x8, STM32F334xx.</li> <li>- (3) Parameter available on all devices except STM32F301x6/8, STM32F318x8, STM32F302xx.</li> <li>-</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• In case of comparator input selected to be connected to IO: GPIO pins are specific to each comparator instance. Refer to description of parameters or to reference manual.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR INMSEL LL_COMP_SetInputHysteresis</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

**LL\_COMP\_SetInputHysteresis**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_COMP_SetInputHysteresis<br/>(COMP_TypeDef * COMPx, uint32_t InputHysteresis)</code>                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Set comparator instance hysteresis mode of the input minus (inverting input).                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> <li>• <b>InputHysteresis:</b> This parameter can be one of the following values: (1) Parameter available only on devices: STM32F302xB/C, STM32F303xB/C, STM32F358xC           <ul style="list-style-type: none"> <li>- LL_COMP_HYSTERESIS_NONE</li> <li>- LL_COMP_HYSTERESIS_LOW (1)</li> <li>- LL_COMP_HYSTERESIS_MEDIUM (1)</li> <li>- LL_COMP_HYSTERESIS_HIGH (1)</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                             |

Reference Manual to  
LL API cross  
reference:

- CSR COMPxHYST LL\_COMP\_SetInputHysteresis

### **LL\_COMP\_GetInputHysteresis**

Function name

**`__STATIC_INLINE uint32_t LL_COMP_GetInputHysteresis(  
(COMP_TypeDef * COMPx)`**

Function description

Get comparator instance hysteresis mode of the minus (inverting) input.

Parameters

- COMPx:** Comparator instance

Return values

- Returned:** value can be one of the following values: (1)  
Parameter available only on devices: STM32F302xB/C,  
STM32F303xB/C, STM32F358xC
  - LL\_COMP\_HYSTERESIS\_NONE
  - LL\_COMP\_HYSTERESIS\_LOW (1)
  - LL\_COMP\_HYSTERESIS\_MEDIUM (1)
  - LL\_COMP\_HYSTERESIS\_HIGH (1)

Reference Manual to  
LL API cross  
reference:

- CSR COMPxHYST LL\_COMP\_GetInputHysteresis

### **LL\_COMP\_SetOutputSelection**

Function name

**`__STATIC_INLINE void LL_COMP_SetOutputSelection(  
(COMP_TypeDef * COMPx, uint32_t OutputSelection)`**

Function description

Set comparator output selection.

Parameters

- COMPx:** Comparator instance
- OutputSelection:** This parameter can be one of the following values: (1) Parameter available on devices: STM32F302x8, STM32F318xx, STM32F303x8, STM32F328xx, STM32F334x8, STM32F302xC, STM32F302xE, STM32F303xC, STM32F358xx, STM32F303xE, STM32F398xx.
  - LL\_COMP\_OUTPUT\_NONE
  - LL\_COMP\_OUTPUT\_TIM1\_BKIN
  - LL\_COMP\_OUTPUT\_TIM1\_BKIN2
  - LL\_COMP\_OUTPUT\_TIM1\_TIM8\_BKIN2
  - LL\_COMP\_OUTPUT\_TIM8\_BKIN (4)
  - LL\_COMP\_OUTPUT\_TIM8\_BKIN2 (4)
  - LL\_COMP\_OUTPUT\_TIM1\_TIM8\_BKIN2 (4)
  - LL\_COMP\_OUTPUT\_TIM20\_BKIN (5)
  - LL\_COMP\_OUTPUT\_TIM20\_BKIN2 (5)
  - LL\_COMP\_OUTPUT\_TIM1\_TIM8\_TIM20\_BKIN2 (5)
  - LL\_COMP\_OUTPUT\_TIM1\_OCCLR\_COMP1\_2\_3\_7 (4)
  - LL\_COMP\_OUTPUT\_TIM2\_OCCLR\_COMP1\_2\_3 (4)
  - LL\_COMP\_OUTPUT\_TIM3\_OCCLR\_COMP1\_2\_4\_5 (4)
  - LL\_COMP\_OUTPUT\_TIM8\_OCCLR\_COMP4\_5\_6\_7 (4)
  - LL\_COMP\_OUTPUT\_TIM3\_OCCLR\_COMP2\_4 (6)

- LL\_COMP\_OUTPUT\_TIM1\_IC1\_COMP2 (2)
- LL\_COMP\_OUTPUT\_TIM2\_IC4\_COMP2 (2)
- LL\_COMP\_OUTPUT\_TIM3\_IC1\_COMP2 (1)
- LL\_COMP\_OUTPUT\_TIM1\_IC1\_COMP1\_2 (3)
- LL\_COMP\_OUTPUT\_TIM2\_IC4\_COMP1\_2 (3)
- LL\_COMP\_OUTPUT\_TIM3\_IC1\_COMP1\_2 (3)
- LL\_COMP\_OUTPUT\_TIM20\_OCCLR\_COMP2 (5)
- LL\_COMP\_OUTPUT\_TIM3\_IC2\_COMP3 (4)
- LL\_COMP\_OUTPUT\_TIM4\_IC1\_COMP3 (4)
- LL\_COMP\_OUTPUT\_TIM15\_IC1\_COMP3 (4)
- LL\_COMP\_OUTPUT\_TIM15\_BKIN
- LL\_COMP\_OUTPUT\_TIM3\_IC3\_COMP4 (1)
- LL\_COMP\_OUTPUT\_TIM4\_IC2\_COMP4
- LL\_COMP\_OUTPUT\_TIM15\_IC2\_COMP4
- LL\_COMP\_OUTPUT\_TIM15\_OCCLR\_COMP4
- LL\_COMP\_OUTPUT\_TIM2\_IC1\_COMP5 (4)
- LL\_COMP\_OUTPUT\_TIM4\_IC3\_COMP5 (4)
- LL\_COMP\_OUTPUT\_TIM17\_IC1\_COMP5 (4)
- LL\_COMP\_OUTPUT\_TIM16\_BKIN
- LL\_COMP\_OUTPUT\_TIM2\_IC2\_COMP6
- LL\_COMP\_OUTPUT\_TIM2\_OCCLR\_COMP6
- LL\_COMP\_OUTPUT\_TIM4\_IC4\_COMP6
- LL\_COMP\_OUTPUT\_TIM16\_IC1\_COMP6
- LL\_COMP\_OUTPUT\_TIM16\_OCCLR\_COMP6
- LL\_COMP\_OUTPUT\_TIM1\_IC2\_COMP7 (4)
- LL\_COMP\_OUTPUT\_TIM2\_IC3\_COMP7 (4)
- LL\_COMP\_OUTPUT\_TIM17\_OCCLR\_COMP7 (4)
- LL\_COMP\_OUTPUT\_TIM17\_BKIN (4)
- (2) Parameter available on devices: STM32F301x8, STM32F302x8, STM32F318xx, STM32F303x8, STM32F328xx, STM32F334x8.
- (3) Parameter available on devices: STM32F302xC, STM32F302xE, STM32F303xC, STM32F358xx, STM32F303xE, STM32F398xx.
- (4) Parameter available on devices: STM32F303xC, STM32F358xx, STM32F303xE, STM32F398xx.
- (5) Parameter available on devices: STM32F303xE, STM32F398xx.
- (6) Parameter available on devices: STM32F303x8, STM32F328xx, STM32F334x8.

**Return values**

- **None**

**Notes**

- Availability of parameters of output selection to timer depends on timers availability on the selected device.

**Reference Manual  
to LL API cross  
reference:**

- CSR COMPxOUTSEL LL\_COMP\_SetOutputSelection

**LL\_COMP\_GetOutputSelection**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_COMP_GetOutputSelection(<br/>COMP_TypeDef * COMPx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Get comparator output selection.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (1)<br/>Parameter available on devices: STM32F302x8,<br/>STM32F318xx, STM32F303x8, STM32F328xx,<br/>STM32F334x8, STM32F302xC, STM32F302xE,<br/>STM32F303xC, STM32F358xx, STM32F303xE,<br/>STM32F398xx. <ul style="list-style-type: none"> <li>- <code>LL_COMP_OUTPUT_NONE</code></li> <li>- <code>LL_COMP_OUTPUT_TIM1_BKIN</code></li> <li>- <code>LL_COMP_OUTPUT_TIM1_BKIN2</code></li> <li>- <code>LL_COMP_OUTPUT_TIM1_TIM8_BKIN2</code></li> <li>- <code>LL_COMP_OUTPUT_TIM8_BKIN (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM8_BKIN2 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM1_TIM8_BKIN2 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM20_BKIN (5)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM20_BKIN2 (5)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM1_TIM8_TIM20_BKIN2 (5)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM1_OCCLR_COMP1_2_3_7 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM2_OCCLR_COMP1_2_3 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM3_OCCLR_COMP1_2_4_5 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM8_OCCLR_COMP4_5_6_7 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM3_OCCLR_COMP2_4 (6)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM1_IC1_COMP2 (2)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM2_IC4_COMP2 (2)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM3_IC1_COMP2 (1)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM1_IC1_COMP1_2 (3)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM2_IC4_COMP1_2 (3)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM3_IC1_COMP1_2 (3)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM20_OCCLR_COMP2 (5)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM3_IC2_COMP3 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM4_IC1_COMP3 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM15_IC1_COMP3 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM15_BKIN</code></li> <li>- <code>LL_COMP_OUTPUT_TIM3_IC3_COMP4 (1)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM4_IC2_COMP4</code></li> <li>- <code>LL_COMP_OUTPUT_TIM15_IC2_COMP4</code></li> <li>- <code>LL_COMP_OUTPUT_TIM15_OCCLR_COMP4</code></li> <li>- <code>LL_COMP_OUTPUT_TIM2_IC1_COMP5 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM4_IC3_COMP5 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM17_IC1_COMP5 (4)</code></li> <li>- <code>LL_COMP_OUTPUT_TIM16_BKIN</code></li> <li>- <code>LL_COMP_OUTPUT_TIM2_IC2_COMP6</code></li> <li>- <code>LL_COMP_OUTPUT_TIM2_OCCLR_COMP6</code></li> <li>- <code>LL_COMP_OUTPUT_TIM4_IC4_COMP6</code></li> <li>- <code>LL_COMP_OUTPUT_TIM16_IC1_COMP6</code></li> <li>- <code>LL_COMP_OUTPUT_TIM16_OCCLR_COMP6</code></li> <li>- <code>LL_COMP_OUTPUT_TIM1_IC2_COMP7 (4)</code></li> </ul> </li> </ul> |

|                                             |                                                                                                                                                                                |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                             | <ul style="list-style-type: none"> <li>- LL_COMP_OUTPUT_TIM2_IC3_COMP7 (4)</li> <li>- LL_COMP_OUTPUT_TIM17_OCCLR_COMP7 (4)</li> <li>- LL_COMP_OUTPUT_TIM17_BKIN (4)</li> </ul> |
|                                             | <ul style="list-style-type: none"> <li>• (2) Parameter available on devices: STM32F301x8, STM32F302x8, STM32F318xx, STM32F303x8, STM32F328xx, STM32F334x8.</li> </ul>          |
|                                             | <ul style="list-style-type: none"> <li>• (3) Parameter available on devices: STM32F302xC, STM32F302xE, STM32F303xC, STM32F358xx, STM32F303xE, STM32F398xx.</li> </ul>          |
|                                             | <ul style="list-style-type: none"> <li>• (4) Parameter available on devices: STM32F303xC, STM32F358xx, STM32F303xE, STM32F398xx.</li> </ul>                                    |
|                                             | <ul style="list-style-type: none"> <li>• (5) Parameter available on devices: STM32F303xE, STM32F398xx.</li> </ul>                                                              |
|                                             | <ul style="list-style-type: none"> <li>• (6) Parameter available on devices: STM32F303x8, STM32F328xx, STM32F334x8.</li> </ul>                                                 |
| Notes                                       | <ul style="list-style-type: none"> <li>• Availability of parameters of output selection to timer depends on timers availability on the selected device.</li> </ul>             |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR COMPxOUTSEL LL_COMP_GetOutputSelection</li> </ul>                                                                                 |

### LL\_COMP\_SetOutputPolarity

|                                             |                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE void LL_COMP_SetOutputPolarity(<br/>COMP_TypeDef * COMPx, uint32_t OutputPolarity)</code>                                                                                                                                                                                      |
| Function description                        | Set comparator instance output polarity.                                                                                                                                                                                                                                                            |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> <li>• <b>OutputPolarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_COMP_OUTPUTPOL_NONINVERTED</li> <li>- LL_COMP_OUTPUTPOL_INVERTED</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                     |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR COMPxPOL LL_COMP_SetOutputPolarity</li> </ul>                                                                                                                                                                                                          |

### LL\_COMP\_GetOutputPolarity

|                                             |                                                                                                                                                                                                                                         |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE uint32_t LL_COMP_GetOutputPolarity(<br/>COMP_TypeDef * COMPx)</code>                                                                                                                                               |
| Function description                        | Get comparator instance output polarity.                                                                                                                                                                                                |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul>                                                                                                                                                   |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_COMP_OUTPUTPOL_NONINVERTED</li> <li>- LL_COMP_OUTPUTPOL_INVERTED</li> </ul> </li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR COMPxPOL LL_COMP_GetOutputPolarity</li> </ul>                                                                                                                                              |

**LL\_COMP\_SetOutputBlankingSource**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_COMP_SetOutputBlankingSource(<br/>COMP_TypeDef * COMPx, uint32_t BlankingSource)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                        | Set comparator instance blanking source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> <li>• <b>BlankingSource:</b> This parameter can be one of the following values: (1) Parameter available on devices: STM32F301x8, STM32F302x8, STM32F318xx, STM32F303x8, STM32F334x8, STM32F328xx. <ul style="list-style-type: none"> <li>- <code>LL_COMP_BLANKINGSRC_NONE</code></li> <li>- <code>LL_COMP_BLANKINGSRC_TIM1_OC5_COMP2</code> (1)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM2_OC3_COMP2</code> (1)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM3_OC3_COMP2</code> (1)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM1_OC5_COMP1_2</code> (2)(3)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM2_OC3_COMP1_2</code> (2)(3)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM3_OC3_COMP1_2</code> (2)(3)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM3_OC4_COMP4</code></li> <li>- <code>LL_COMP_BLANKINGSRC_TIM15_OC1_COMP4</code></li> <li>- <code>LL_COMP_BLANKINGSRC_TIM2_OC4_COMP6</code> (2)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM15_OC2_COMP6</code> (1)(2)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM1_OC5_COMP1_2_7</code> (3)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM2_OC4_COMP3_6</code> (3)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM8_OC5_COMP4_5_6_7</code> (3)</li> <li>- <code>LL_COMP_BLANKINGSRC_TIM15_OC2_COMP6_7</code> (3)</li> </ul> </li> <li>• (2) Parameter available on devices: STM32F302xE, STM32F302xC.</li> <li>• (3) Parameter available on devices: STM32F303xE, STM32F398xx, STM32F303xC, STM32F358xx.</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Notes                                       | <ul style="list-style-type: none"> <li>• Blanking source may be specific to each comparator instance. Refer to description of parameters or to reference manual.</li> <li>• Availability of parameters of blanking source from timer depends on timers availability on the selected device.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR COMPxBLANKING LL_COMP_SetOutputBlankingSource</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**LL\_COMP\_GetOutputBlankingSource**

|                      |                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_COMP_GetOutputBlankingSource(<br/>COMP_TypeDef * COMPx)</code>                                                                                                                                 |
| Function description | Get comparator instance blanking source.                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul>                                                                                                                                            |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (1) Parameter available on devices: STM32F301x8, STM32F302x8, STM32F318xx, STM32F303x8, STM32F334x8, STM32F328xx.</li> </ul> |

- LL\_COMP\_BLANKINGSRC\_NONE
- LL\_COMP\_BLANKINGSRC\_TIM1\_OC5\_COMP2 (1)
- LL\_COMP\_BLANKINGSRC\_TIM2\_OC3\_COMP2 (1)
- LL\_COMP\_BLANKINGSRC\_TIM3\_OC3\_COMP2 (1)
- LL\_COMP\_BLANKINGSRC\_TIM1\_OC5\_COMP1\_2 (2)(3)
- LL\_COMP\_BLANKINGSRC\_TIM2\_OC3\_COMP1\_2 (2)(3)
- LL\_COMP\_BLANKINGSRC\_TIM3\_OC3\_COMP1\_2 (2)(3)
- LL\_COMP\_BLANKINGSRC\_TIM3\_OC4\_COMP4
- LL\_COMP\_BLANKINGSRC\_TIM15\_OC1\_COMP4
- LL\_COMP\_BLANKINGSRC\_TIM2\_OC4\_COMP6 (2)
- LL\_COMP\_BLANKINGSRC\_TIM15\_OC2\_COMP6 (1)(2)
- LL\_COMP\_BLANKINGSRC\_TIM1\_OC5\_COMP1\_2\_7 (3)
- LL\_COMP\_BLANKINGSRC\_TIM2\_OC4\_COMP3\_6 (3)
- LL\_COMP\_BLANKINGSRC\_TIM8\_OC5\_COMP4\_5\_6\_7 (3)
- LL\_COMP\_BLANKINGSRC\_TIM15\_OC2\_COMP6\_7 (3)
- (2) Parameter available on devices: STM32F302xE,  
STM32F302xC.
- (3) Parameter available on devices: STM32F303xE,  
STM32F398xx, STM32F303xC, STM32F358xx.

- |                                                      |                                                                                                                                                                                                                                                                                                        |
|------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                                | <ul style="list-style-type: none"> <li>• Availability of parameters of blanking source from timer depends on timers availability on the selected device.</li> <li>• Blanking source may be specific to each comparator instance. Refer to description of parameters or to reference manual.</li> </ul> |
| Reference<br>Manual to LL<br>API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR COMPxBLANKING LL_COMP_GetOutputBlankingSource</li> </ul>                                                                                                                                                                                                  |

### LL\_COMP\_Enable

- |                                                   |                                                                                                                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_COMP_Enable (COMP_TypeDef * COMPx)</code>                                                                                                                               |
| Function description                              | Enable comparator instance.                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul>                                                                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• After enable from off state, comparator requires a delay to reach propagation delay specification. Refer to device datasheet, parameter "tSTART".</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR COMPxEN LL_COMP_Enable</li> </ul>                                                                                                                        |

**LL\_COMP\_Disable**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_COMP_Disable (COMP_TypeDef * COMPx)</code>              |
| Function description                              | Disable comparator instance.                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR COMPxEN LL_COMP_Disable</li> </ul>       |

**LL\_COMP\_IsEnabled**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_COMP_IsEnabled (COMP_TypeDef * COMPx)</code>        |
| Function description                              | Get comparator enable state (0: COMP is disabled, 1: COMP is enabled)                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR COMPxEN LL_COMP_IsEnabled</li> </ul>     |

**LL\_COMP\_Lock**

|                                                   |                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_COMP_Lock (COMP_TypeDef * COMPx)</code>                                                                                                                                |
| Function description                              | Lock comparator instance.                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul>                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• Once locked, comparator configuration can be accessed in read-only.</li> <li>• The only way to unlock the comparator is a device hardware reset.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR COMPxLOCK LL_COMP_Lock</li> </ul>                                                                                                                       |

**LL\_COMP\_IsLocked**

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_COMP_IsLocked (COMP_TypeDef * COMPx)</code>         |
| Function description | Get comparator lock state (0: COMP is unlocked, 1: COMP is locked).                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> Comparator instance</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>    |

---

|                                             |                                                                                                                                                                                                  |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>Once locked, comparator configuration can be accessed in read-only.</li> <li>The only way to unlock the comparator is a device hardware reset.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CSR COMPxLOCK LL_COMP_IsLocked</li> </ul>                                                                                                                 |

### LL\_COMP\_ReadOutputLevel

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_COMP_ReadOutputLevel(<br/>COMP_TypeDef * COMPx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description                        | Read comparator instance output level.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>COMPx:</b> Comparator instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Return values                               | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>LL_COMP_OUTPUT_LEVEL_LOW</li> <li>LL_COMP_OUTPUT_LEVEL_HIGH</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                |
| Notes                                       | <ul style="list-style-type: none"> <li>The comparator output level depends on the selected polarity (Refer to function LL_COMP_SetOutputPolarity()). If the comparator polarity is not inverted: Comparator output is low when the input plus is at a lower voltage than the input minusComparator output is high when the input plus is at a higher voltage than the input minus If the comparator polarity is inverted:Comparator output is high when the input plus is at a lower voltage than the input minusComparator output is low when the input plus is at a higher voltage than the input minus</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CSR COMPxOUT LL_COMP_ReadOutputLevel</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

### LL\_COMP\_DeInit

|                      |                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_COMP_DeInit (COMP_TypeDef * COMPx)</code>                                                                                                                                                                                            |
| Function description | De-initialize registers of the selected COMP instance to their default reset values.                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>COMPx:</b> COMP instance</li> </ul>                                                                                                                                                                             |
| Return values        | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>SUCCESS: COMP registers are de-initialized</li> <li>ERROR: COMP registers are not de-initialized</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>If comparator is locked, de-initialization by software is not possible. The only way to unlock the comparator is a device hardware reset.</li> </ul>                                                               |

**LL\_COMP\_Init**

|                      |                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_COMP_Init (COMP_TypeDef * COMPx,<br/>LL_COMP_InitTypeDef * COMP_InitStruct)</b>                                                                                                                                                                                        |
| Function description | Initialize some features of COMP instance.                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>COMPx:</b> COMP instance</li> <li>• <b>COMP_InitStruct:</b> Pointer to a LL_COMP_InitTypeDef structure</li> </ul>                                                                                                                            |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:             <ul style="list-style-type: none"> <li>– SUCCESS: COMP registers are initialized</li> <li>– ERROR: COMP registers are not initialized</li> </ul> </li> </ul>                              |
| Notes                | <ul style="list-style-type: none"> <li>• This function configures features of the selected COMP instance. Some features are also available at scope COMP common instance (common to several COMP instances). Refer to functions having argument "COMPxy_COMMON" as parameter.</li> </ul> |

**LL\_COMP\_StructInit**

|                      |                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_COMP_StructInit (LL_COMP_InitTypeDef *<br/>COMP_InitStruct)</b>                                                                                         |
| Function description | Set each LL_COMP_InitTypeDef field to default value.                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>COMP_InitStruct:</b> pointer to a LL_COMP_InitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                    |

## 61.3 COMP Firmware driver defines

### 61.3.1 COMP

*Comparator common modes - Window mode*

|                                                   |                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>LL_COMP_WINDOWMODE_DISABLE</b>                 | Window mode disable:<br>Comparators 1 and 2<br>are independent                                                                                                                                                  |
| <b>LL_COMP_WINDOWMODE_COMP1_INPUT_PLUS_COMMON</b> | Window mode enable:<br>Comparators instances<br>pair COMP1 and<br>COMP2 have their<br>input plus connected<br>together. The common<br>input is COMP1 input<br>plus (COMP2 input plus<br>is no more accessible). |
| <b>LL_COMP_WINDOWMODE_COMP3_INPUT_PLUS_COMMON</b> | Window mode enable:<br>Comparators instances<br>pair COMP3 and<br>COMP4 have their<br>input plus connected<br>together. The common                                                                              |

|                                            |                                                                                                                                                                                         |
|--------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                            | input is COMP3 input plus (COMP4 input plus is no more accessible).                                                                                                                     |
| LL_COMP_WINDOWMODE_COMP5_INPUT_PLUS_COMMON | Window mode enable: Comparators instances pair COMP5 and COMP6 have their input plus connected together. The common input is COMP5 input plus (COMP6 input plus is no more accessible). |

**Definitions of COMP hardware constraints delays**

|                                      |                                                  |
|--------------------------------------|--------------------------------------------------|
| LL_COMP_DELAY_STARTUP_US             | Delay for COMP startup time                      |
| LL_COMP_DELAY_VOLTAGE_SCALER_STAB_US | Delay for COMP voltage scaler stabilization time |

**Comparator input - Hysteresis**

|                         |                                                                                 |
|-------------------------|---------------------------------------------------------------------------------|
| LL_COMP_HYSTESIS_NONE   | No hysteresis                                                                   |
| LL_COMP_HYSTESIS_LOW    | Hysteresis level low (available only on devices: STM32F303xB/C, STM32F358xC)    |
| LL_COMP_HYSTESIS_MEDIUM | Hysteresis level medium (available only on devices: STM32F303xB/C, STM32F358xC) |
| LL_COMP_HYSTESIS_HIGH   | Hysteresis level high (available only on devices: STM32F303xB/C, STM32F358xC)   |

**Comparator inputs - Input minus (input inverting) selection**

|                                |                                                                                                                                                                                                                          |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_COMP_INPUT_MINUS_1_4VREFINT | Comparator input minus connected to 1/4 VrefInt                                                                                                                                                                          |
| LL_COMP_INPUT_MINUS_1_2VREFINT | Comparator input minus connected to 1/2 VrefInt                                                                                                                                                                          |
| LL_COMP_INPUT_MINUS_3_4VREFINT | Comparator input minus connected to 3/4 VrefInt                                                                                                                                                                          |
| LL_COMP_INPUT_MINUS_VREFINT    | Comparator input minus connected to VrefInt                                                                                                                                                                              |
| LL_COMP_INPUT_MINUS_DAC1_CH1   | Comparator input minus connected to DAC1 channel 1 (DAC_OUT1)                                                                                                                                                            |
| LL_COMP_INPUT_MINUS_DAC1_CH2   | Comparator input minus connected to DAC1 channel 2 (DAC_OUT2)                                                                                                                                                            |
| LL_COMP_INPUT_MINUS_IO1        | Comparator input minus connected to IO1 (pin PA0 for COMP1, pin PA2 for COMP2, PD15 for COMP3, PE8 for COMP4, PD13 for COMP5, PD10 for COMP6, PC0 for COMP7 (COMP instance availability depends on the selected device)) |
| LL_COMP_INPUT_MINUS_IO2        | Comparator input minus connected to IO2 (PB12 for COMP3, PB2 for COMP4, PB10 for COMP5, PB15 for COMP6 (COMP instance availability depends on the selected device))                                                      |

|                                                                       |                                                                                                                                                                                                                                                                                          |
|-----------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_COMP_INPUT_MINUS_IO3                                               | Comparator input minus connected to IO3 (pin PA5 for COMP1/2/3/4/5/6/7 (COMP instance availability depends on the selected device))                                                                                                                                                      |
| LL_COMP_INPUT_MINUS_IO4                                               | Comparator input minus connected to IO4 (pin PA4 for COMP1/2/3/4/5/6/7 (COMP instance availability depends on the selected device))                                                                                                                                                      |
| <b>Comparator inputs - Input plus (input non-inverting) selection</b> |                                                                                                                                                                                                                                                                                          |
| LL_COMP_INPUT_PLUS_IO1                                                | Comparator input plus connected to IO1 (pin PA7 for COMP2, PB14 for COMP3, PB0 for COMP4, PD12 for COMP5, PD11 for COMP6, PA0 for COMP7) (COMP instance availability depends on the selected device)                                                                                     |
| LL_COMP_INPUT_PLUS_IO2                                                | Comparator input plus connected to IO2 (pin PA3 for COMP2, PD14 for COMP3, PE7 for COMP4, PB13 for COMP5, PB11 for COMP6, PC1 for COMP7) (COMP instance availability depends on the selected device)                                                                                     |
| LL_COMP_INPUT_PLUS_DAC1_CH1_COMP1                                     | Comparator input plus connected to DAC1 channel 1 (DAC_OUT1), through dedicated switch (Note: this switch is solely intended to redirect signals onto high impedance input, such as COMP1 input plus (highly resistive switch)) (specific to COMP instance: COMP1)                       |
| LL_COMP_INPUT_PLUS_DAC1_CH1                                           | Comparator input plus connected to DAC1 channel 1 (DAC_OUT1), through dedicated switch. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| <b>Comparator output - Blanking source</b>                            |                                                                                                                                                                                                                                                                                          |
| LL_COMP_BLANKINGSRC_NONE                                              | Comparator output without blanking                                                                                                                                                                                                                                                       |
| LL_COMP_BLANKINGSRC_TIM1_OC5_COMP1_2_7                                | Comparator output blanking source TIM1 OC5 (specific to COMP instance: COMP1, COMP2, COMP7)                                                                                                                                                                                              |
| LL_COMP_BLANKINGSRC_TIM2_OC3_COMP1_2                                  | Comparator output blanking source TIM2 OC3 (specific to COMP instance: COMP1, COMP2)                                                                                                                                                                                                     |
| LL_COMP_BLANKINGSRC_TIM3_OC3_COMP1_2                                  | Comparator output blanking source TIM3 OC3 (specific to COMP instance: COMP1, COMP2)                                                                                                                                                                                                     |

|                                          |                                                                                                                                                                                                                                              |
|------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_COMP_BLANKINGSRC_TIM2_OC4_COMP3_6     | Comparator output blanking source TIM2 OC4 (specific to COMP instance: COMP3, COMP6)                                                                                                                                                         |
| LL_COMP_BLANKINGSRC_TIM8_OC5_COMP4_5_6_7 | Comparator output blanking source TIM8 OC5 (specific to COMP instance: COMP4, COMP5, COMP6, COMP7)                                                                                                                                           |
| LL_COMP_BLANKINGSRC_TIM15_OC2_COMP6_7    | Comparator output blanking source TIM15 OC2 (specific to COMP instance: COMP6, COMP7)                                                                                                                                                        |
| LL_COMP_BLANKINGSRC_TIM3_OC4_COMP4       | Comparator output blanking source TIM3 OC4 (specific to COMP instance: COMP4)                                                                                                                                                                |
| LL_COMP_BLANKINGSRC_TIM15_OC1_COMP4      | Comparator output blanking source TIM15 OC1 (specific to COMP instance: COMP4)                                                                                                                                                               |
| LL_COMP_BLANKINGSRC_TIM1_OC5             | Comparator output blanking source TIM1 OC5. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| LL_COMP_BLANKINGSRC_TIM2_OC3             | Comparator output blanking source TIM2 OC3. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| LL_COMP_BLANKINGSRC_TIM2_OC4             | Comparator output blanking source TIM2 OC4. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| LL_COMP_BLANKINGSRC_TIM3_OC3             | Comparator output blanking source TIM3 OC3. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |

|                                            |                                                                                                                                                                                                                                               |
|--------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>LL_COMP_BLANKINGSRC_TIM3_OC4</code>  | Comparator output blanking source TIM3 OC4. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.  |
| <code>LL_COMP_BLANKINGSRC_TIM8_OC5</code>  | Comparator output blanking source TIM8 OC5. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.  |
| <code>LL_COMP_BLANKINGSRC_TIM15_OC1</code> | Comparator output blanking source TIM15 OC1. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| <code>LL_COMP_BLANKINGSRC_TIM15_OC2</code> | Comparator output blanking source TIM15 OC2. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |

#### ***Comparator output - Output level***

|                                        |                                                                                              |
|----------------------------------------|----------------------------------------------------------------------------------------------|
| <code>LL_COMP_OUTPUT_LEVEL_LOW</code>  | Comparator output level low (if the polarity is not inverted, otherwise to be complemented)  |
| <code>LL_COMP_OUTPUT_LEVEL_HIGH</code> | Comparator output level high (if the polarity is not inverted, otherwise to be complemented) |

#### ***Comparator output - Output polarity***

|                                            |                                                                                                                                                             |
|--------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>LL_COMP_OUTPUTPOL_NONINVERTED</code> | COMP output polarity is not inverted: comparator output is high when the plus (non-inverting) input is at a higher voltage than the minus (inverting) input |
| <code>LL_COMP_OUTPUTPOL_INVERTED</code>    | COMP output polarity is inverted: comparator output is low when the plus (non-inverting) input is at a lower voltage than the minus (inverting) input       |

#### ***Comparator output - Output selection***

|                                  |                                                                                         |
|----------------------------------|-----------------------------------------------------------------------------------------|
| <code>LL_COMP_OUTPUT_NONE</code> | COMP output is not connected to other peripherals (except GPIO and EXTI that are always |
|----------------------------------|-----------------------------------------------------------------------------------------|

|                                       |                                                                                                            |
|---------------------------------------|------------------------------------------------------------------------------------------------------------|
|                                       | connected to COMP output<br>(specific to COMP instance:<br>COMP2)                                          |
| LL_COMP_OUTPUT_TIM1_BKIN              | COMP output connected to TIM1<br>break input (BKIN)                                                        |
| LL_COMP_OUTPUT_TIM1_BKIN2             | COMP output connected to TIM1<br>break input 2 (BKIN2)                                                     |
| LL_COMP_OUTPUT_TIM8_BKIN              | COMP output connected to TIM8<br>break input (BKIN)                                                        |
| LL_COMP_OUTPUT_TIM8_BKIN2             | COMP output connected to TIM8<br>break input 2 (BKIN2)                                                     |
| LL_COMP_OUTPUT_TIM1_TIM8_BKIN2        | COMP output connected to TIM1<br>break input 2 and TIM8 break input<br>2 (BKIN2)                           |
| LL_COMP_OUTPUT_TIM1_OCCLR_COMP1_2_3_7 | COMP output connected to TIM1<br>OCREF clear (specific to COMP<br>instance: COMP1, COMP2,<br>COMP3, COMP7) |
| LL_COMP_OUTPUT_TIM2_OCCLR_COMP1_2_3   | COMP output connected to TIM2<br>OCREF clear (specific to COMP<br>instance: COMP1, COMP2,<br>COMP3)        |
| LL_COMP_OUTPUT_TIM3_OCCLR_COMP1_2_4_5 | COMP output connected to TIM3<br>OCREF clear (specific to COMP<br>instance: COMP1, COMP2,<br>COMP4, COMP5) |
| LL_COMP_OUTPUT_TIM8_OCCLR_COMP4_5_6_7 | COMP output connected to TIM8<br>OCREF clear (specific to COMP<br>instance: COMP4, COMP5,<br>COMP6, COMP7) |
| LL_COMP_OUTPUT_TIM1_IC1_COMP1_2       | COMP output connected to TIM1<br>input capture 1 (specific to COMP<br>instance: COMP2)                     |
| LL_COMP_OUTPUT_TIM2_IC4_COMP1_2       | COMP output connected to TIM2<br>input capture 4 (specific to COMP<br>instance: COMP2)                     |
| LL_COMP_OUTPUT_TIM3_IC1_COMP1_2       | COMP output connected to TIM3<br>input capture 1 (specific to COMP<br>instance: COMP2)                     |
| LL_COMP_OUTPUT_TIM3_IC2_COMP3         | COMP output connected to TIM3<br>input capture 2 (specific to COMP<br>instance: COMP3)                     |
| LL_COMP_OUTPUT_TIM4_IC1_COMP3         | COMP output connected to TIM4<br>input capture 1 (specific to COMP<br>instance: COMP3)                     |
| LL_COMP_OUTPUT_TIM15_IC1_COMP3        | COMP output connected to TIM15<br>input capture 1 (specific to COMP                                        |

|                                  |                                                                                   |
|----------------------------------|-----------------------------------------------------------------------------------|
| LL_COMP_OUTPUT_TIM15_BKIN_COMP3  | instance: COMP3)<br>COMP output connected to TIM15 break input (BKIN)             |
| LL_COMP_OUTPUT_TIM3_IC3_COMP4    | COMP output connected to TIM3 input capture 3 (specific to COMP instance: COMP4)  |
| LL_COMP_OUTPUT_TIM4_IC2_COMP4    | COMP output connected to TIM4 input capture 2 (specific to COMP instance: COMP4)  |
| LL_COMP_OUTPUT_TIM15_IC2_COMP4   | COMP output connected to TIM15 input capture 1 (specific to COMP instance: COMP4) |
| LL_COMP_OUTPUT_TIM15_OCCLR_COMP4 | COMP output connected to TIM15 OCREF clear (specific to COMP instance: COMP4)     |
| LL_COMP_OUTPUT_TIM2_IC1_COMP5    | COMP output connected to TIM2 input capture 1 (specific to COMP instance: COMP5)  |
| LL_COMP_OUTPUT_TIM4_IC3_COMP5    | COMP output connected to TIM4 input capture 3 (specific to COMP instance: COMP5)  |
| LL_COMP_OUTPUT_TIM17_IC1_COMP5   | COMP output connected to TIM17 input capture 1 (specific to COMP instance: COMP5) |
| LL_COMP_OUTPUT_TIM16_BKIN_COMP5  | COMP output connected to TIM16 break input (BKIN)                                 |
| LL_COMP_OUTPUT_TIM2_IC2_COMP6    | COMP output connected to TIM2 input capture 2 (specific to COMP instance: COMP6)  |
| LL_COMP_OUTPUT_TIM2_OCCLR_COMP6  | COMP output connected to TIM2 OCREF clear (specific to COMP instance: COMP6)      |
| LL_COMP_OUTPUT_TIM4_IC4_COMP6    | COMP output connected to TIM4 input capture 4 (specific to COMP instance: COMP6)  |
| LL_COMP_OUTPUT_TIM16_IC1_COMP6   | COMP output connected to TIM16 input capture 1 (specific to COMP instance: COMP6) |
| LL_COMP_OUTPUT_TIM16_OCCLR_COMP6 | COMP output connected to TIM16 OCREF clear (specific to COMP instance: COMP6)     |
| LL_COMP_OUTPUT_TIM1_IC2_COMP7    | COMP output connected to TIM2 input capture 1 (specific to COMP instance: COMP7)  |
| LL_COMP_OUTPUT_TIM2_IC3_COMP7    | COMP output connected to TIM4 input capture 3 (specific to COMP instance: COMP7)  |

|                                  |                                                                                                                                                                                                                                                 |
|----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_COMP_OUTPUT_TIM17_OCCLR_COMP7 | COMP output connected to TIM17 OCREF clear (specific to COMP instance: COMP7)                                                                                                                                                                   |
| LL_COMP_OUTPUT_TIM17_BKIN_COMP7  | COMP output connected to TIM17 break input (BKIN)                                                                                                                                                                                               |
| LL_COMP_OUTPUT_TIM1_IC1          | COMP output connected to TIM1 input capture 1. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| LL_COMP_OUTPUT_TIM1_IC2          | COMP output connected to TIM2 input capture 1. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| LL_COMP_OUTPUT_TIM1_OCCLR        | COMP output connected to TIM1 OCREF clear. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.     |
| LL_COMP_OUTPUT_TIM2_IC1          | COMP output connected to TIM2 input capture 1. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| LL_COMP_OUTPUT_TIM2_IC2          | COMP output connected to TIM2 input capture 2. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| LL_COMP_OUTPUT_TIM2_IC3          | COMP output connected to TIM4 input capture 3. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for                            |

LL\_COMP\_OUTPUT\_TIM2\_IC4

COMP instance constraints.

COMP output connected to TIM2 input capture 4. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM3\_IC1

COMP output connected to TIM3 input capture 1. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM3\_IC2

COMP output connected to TIM3 input capture 2. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM3\_IC3

COMP output connected to TIM3 input capture 3. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM3\_OCCLR

COMP output connected to TIM3 OCREF clear. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM4\_IC1

COMP output connected to TIM4 input capture 1. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM4\_IC2

COMP output connected to TIM4 input capture 2. Caution: Parameter specific to COMP

|                           |                                                                                                                                                                                                                                                  |
|---------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                           | instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.                                                                                     |
| LL_COMP_OUTPUT_TIM4_IC3   | COMP output connected to TIM4 input capture 3. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.  |
| LL_COMP_OUTPUT_TIM4_IC4   | COMP output connected to TIM4 input capture 4. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.  |
| LL_COMP_OUTPUT_TIM8_OCCLR | COMP output connected to TIM8 OCREF clear. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.      |
| LL_COMP_OUTPUT_TIM15_IC1  | COMP output connected to TIM15 input capture 1. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| LL_COMP_OUTPUT_TIM15_IC2  | COMP output connected to TIM15 input capture 1. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
| LL_COMP_OUTPUT_TIM15_BKIN | COMP output connected to TIM15 break input (BKIN). Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for                         |

LL\_COMP\_OUTPUT\_TIM15\_OCCLR

COMP instance constraints.

COMP output connected to TIM15 OCREF clear. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM16\_IC1

COMP output connected to TIM16 input capture 1. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM16\_BKIN

COMP output connected to TIM16 break input (BKIN). Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM16\_OCCLR

COMP output connected to TIM16 OCREF clear. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM17\_IC1

COMP output connected to TIM17 input capture 1. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

LL\_COMP\_OUTPUT\_TIM17\_BKIN

COMP output connected to TIM17 break input (BKIN). Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints.

|                            |                                                                                                                                                                                                                                              |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_COMP_OUTPUT_TIM17_OCCLR | COMP output connected to TIM17 OCREF clear. Caution: Parameter specific to COMP instances, defined with generic naming, not taking into account COMP instance constraints. Refer to literal definitions above for COMP instance constraints. |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**Comparator modes - Power mode**

|                                 |                                    |
|---------------------------------|------------------------------------|
| LL_COMP_POWERMODE_HIGHSPEED     | COMP power mode to high speed      |
| LL_COMP_POWERMODE_MEDIUMSPEED   | COMP power mode to medium speed    |
| LL_COMP_POWERMODE_LOWPOWER      | COMP power mode to low power       |
| LL_COMP_POWERMODE_ULTRALOWPOWER | COMP power mode to ultra-low power |

**COMP helper macro**

|                                |                     |
|--------------------------------|---------------------|
| <u>LL_COMP_COMMON_INSTANCE</u> | <b>Description:</b> |
|--------------------------------|---------------------|

- Helper macro to select the COMP common instance to which is belonging the selected COMP instance.

**Parameters:**

- COMPx: COMP instance

**Return value:**

- COMP: common instance or value "0" if there is no COMP common instance.

**Notes:**

- COMP common register instance can be used to set parameters common to several COMP instances. Refer to functions having argument "COMPxy\_COMMON" as parameter.

**Common write and read registers macro**

|                         |                     |
|-------------------------|---------------------|
| <u>LL_COMP_WriteReg</u> | <b>Description:</b> |
|-------------------------|---------------------|

- Write a value in COMP register.

**Parameters:**

- INSTANCE: comparator instance
- REG: Register to be written
- VALUE: Value to be written in the register

**Return value:**

- None

**LL\_COMP\_ReadReg****Description:**

- Read a value in COMP register.

**Parameters:**

- \_\_INSTANCE\_\_: comparator instance
- \_\_REG\_\_: Register to be read

**Return value:**

- Register: value

## 62 LL CORTEX Generic Driver

### 62.1 CORTEX Firmware driver API description

#### 62.1.1 Detailed description of functions

##### **LL\_SYSTICK\_IsActiveCounterFlag**

Function name **`__STATIC_INLINE uint32_t LL_SYSTICK_IsActiveCounterFlag(void)`**

Function description This function checks if the Systick counter flag is active or not.

Return values • **State:** of bit (1 or 0).

Notes • It can be used in timeout function on application side.

Reference Manual to  
LL API cross  
reference: • STK\_CTRL COUNTFLAG LL\_SYSTICK\_IsActiveCounterFlag

##### **LL\_SYSTICK\_SetClkSource**

Function name **`__STATIC_INLINE void LL_SYSTICK_SetClkSource(uint32_t Source)`**

Function description Configures the SysTick clock source.

Parameters • **Source:** This parameter can be one of the following values:  
– LL\_SYSTICK\_CLKSOURCE\_HCLK\_DIV8  
– LL\_SYSTICK\_CLKSOURCE\_HCLK

Return values • **None**

Reference Manual to  
LL API cross  
reference: • STK\_CTRL CLKSOURCE LL\_SYSTICK\_SetClkSource

##### **LL\_SYSTICK\_GetClkSource**

Function name **`__STATIC_INLINE uint32_t LL_SYSTICK_GetClkSource(void)`**

Function description Get the SysTick clock source.

Return values • **Returned:** value can be one of the following values:  
– LL\_SYSTICK\_CLKSOURCE\_HCLK\_DIV8  
– LL\_SYSTICK\_CLKSOURCE\_HCLK

Reference Manual to  
LL API cross  
reference: • STK\_CTRL CLKSOURCE LL\_SYSTICK\_GetClkSource

**LL\_SYSTICK\_EnableIT**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSTICK_EnableIT (void)</code>                           |
| Function description                              | Enable SysTick exception request.                                                      |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• STK_CTRL TICKINT LL_SYSTICK_EnableIT</li></ul> |

**LL\_SYSTICK\_DisableIT**

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSTICK_DisableIT (void)</code>                           |
| Function description                              | Disable SysTick exception request.                                                      |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• STK_CTRL TICKINT LL_SYSTICK_DisableIT</li></ul> |

**LL\_SYSTICK\_IsEnabledIT**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SYSTICK_IsEnabledIT (void)</code>                       |
| Function description                              | Checks if the SYSTICK interrupt is enabled or disabled.                                   |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• STK_CTRL TICKINT LL_SYSTICK_IsEnabledIT</li></ul> |

**LL\_LPM\_EnableSleep**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_LPM_EnableSleep (void)</code>                            |
| Function description                              | Processor uses sleep as its low power mode.                                            |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• SCB_SCR SLEEPDEEP LL_LPM_EnableSleep</li></ul> |

**LL\_LPM\_EnableDeepSleep**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_LPM_EnableDeepSleep (void)</code>                            |
| Function description                              | Processor uses deep sleep as its low power mode.                                           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• SCB_SCR SLEEPDEEP LL_LPM_EnableDeepSleep</li></ul> |

**LL\_LPM\_EnableSleepOnExit**

|                                                   |                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_LPM_EnableSleepOnExit (void )</code></b>                                                                                          |
| Function description                              | Configures sleep-on-exit when returning from Handler mode to Thread mode.                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Setting this bit to 1 enables an interrupt-driven application to avoid returning to an empty main application.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SCB_SCR SLEEPONEXIT LL_LPM_EnableSleepOnExit</li> </ul>                                                                   |

**LL\_LPM\_DisableSleepOnExit**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_LPM_DisableSleepOnExit (void )</code></b>                        |
| Function description                              | Do not sleep when returning to Thread mode.                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SCB_SCR SLEEPONEXIT LL_LPM_DisableSleepOnExit</li> </ul> |

**LL\_LPM\_EnableEventOnPend**

|                                                   |                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_LPM_EnableEventOnPend (void )</code></b>                       |
| Function description                              | Enabled events and all interrupts, including disabled interrupts, can wakeup the processor.     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SCB_SCR SEVEONPEND LL_LPM_EnableEventOnPend</li> </ul> |

**LL\_LPM\_DisableEventOnPend**

|                                                   |                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_LPM_DisableEventOnPend (void )</code></b>                       |
| Function description                              | Only enabled interrupts or events can wakeup the processor, disabled interrupts are excluded.    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SCB_SCR SEVEONPEND LL_LPM_DisableEventOnPend</li> </ul> |

**LL\_HANDLER\_EnableFault**

|                      |                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_HANDLER_EnableFault (uint32_t Fault)</code></b>                                       |
| Function description | Enable a fault in System handler control register (SHCSR)                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Fault:</b> This parameter can be a combination of the following</li> </ul> |

---

|                                                   |                                                |
|---------------------------------------------------|------------------------------------------------|
|                                                   | values:                                        |
|                                                   | – LL_HANDLER_FAULT_USG                         |
|                                                   | – LL_HANDLER_FAULT_BUS                         |
|                                                   | – LL_HANDLER_FAULT_MEM                         |
| Return values                                     | • <b>None</b>                                  |
| Reference Manual to<br>LL API cross<br>reference: | • SCB_SHCSR MEMFAULTENA LL_HANDLER_EnableFault |

### **LL\_HANDLER\_DisableFault**

|                                                   |                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HANDLER_DisableFault (uint32_t Fault)</code></b>                                                                                                                                                                                                   |
| Function description                              | Disable a fault in System handler control register (SHCSR)                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Fault:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>– LL_HANDLER_FAULT_USG</li> <li>– LL_HANDLER_FAULT_BUS</li> <li>– LL_HANDLER_FAULT_MEM</li> </ul> </li> </ul> |
| Return values                                     | • <b>None</b>                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | • SCB_SHCSR MEMFAULTENA LL_HANDLER_DisableFault                                                                                                                                                                                                                                    |

### **LL\_CPUID\_GetImplementer**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_CPUID_GetImplementer (void )</code></b>                       |
| Function description                              | Get Implementer code.                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> should be equal to 0x41 for ARM</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | • SCB_CPUID IMPLEMENTER LL_CPUID_GetImplementer                                                   |

### **LL\_CPUID\_GetVariant**

|                                                   |                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_CPUID_GetVariant (void )</code></b>                               |
| Function description                              | Get Variant number (The r value in the rnpn product revision identifier)                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between 0 and 255 (0x0: revision 0)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | • SCB_CPUID VARIANT LL_CPUID_GetVariant                                                               |

**LL\_CPUID\_GetConstant**

|                                                   |                                                                                                                |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_CPUID_GetConstant (void )</code></b>                                      |
| Function description                              | Get Constant number.                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> should be equal to 0xF for Cortex-M4 devices</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SCB_CPUID ARCHITECTURE LL_CPUID_GetConstant</li> </ul>                |

**LL\_CPUID\_GetParNo**

|                                                   |                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_CPUID_GetParNo (void )</code></b>                                   |
| Function description                              | Get Part number.                                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> should be equal to 0xC24 for Cortex-M4</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SCB_CPUID PARTNO LL_CPUID_GetParNo</li> </ul>                   |

**LL\_CPUID\_GetRevision**

|                                                   |                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_CPUID_GetRevision (void )</code></b>                          |
| Function description                              | Get Revision number (The p value in the rpn product revision identifier, indicates patch release)  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between 0 and 255 (0x1: patch 1)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SCB_CPUID REVISION LL_CPUID_GetRevision</li> </ul>        |

**LL\_MPU\_Enable**

|                                                   |                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_MPU_Enable (uint32_t Options)</code></b>                                                                                                                                                                                                                                                      |
| Function description                              | Enable MPU with input options.                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Options:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_MPU_CTRL_HFNMI_PRIVDEF_NONE</li> <li>- LL_MPU_CTRL_HARDFAULT_NMI</li> <li>- LL_MPU_CTRL_PRIVILEGED_DEFAULT</li> <li>- LL_MPU_CTRL_HFNMI_PRIVDEF</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MPU_CTRL_ENABLE LL_MPU_Enable</li> </ul>                                                                                                                                                                                                                                              |

**LL\_MPU\_Disable**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_MPU_Disable (void )</code></b>                    |
| Function description                              | Disable MPU.                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MPU_CTRL ENABLE LL_MPU_Disable</li> </ul> |

**LL\_MPU\_IsEnabled**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_MPU_IsEnabled (void )</code></b>                |
| Function description                              | Check if MPU is enabled or not.                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MPU_CTRL ENABLE LL_MPU_IsEnabled</li> </ul> |

**LL\_MPU\_EnableRegion**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_MPU_EnableRegion (uint32_t Region)</code></b>                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Enable a MPU region.                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Region:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_MPU_REGION_NUMBER0</li> <li>- LL_MPU_REGION_NUMBER1</li> <li>- LL_MPU_REGION_NUMBER2</li> <li>- LL_MPU_REGION_NUMBER3</li> <li>- LL_MPU_REGION_NUMBER4</li> <li>- LL_MPU_REGION_NUMBER5</li> <li>- LL_MPU_REGION_NUMBER6</li> <li>- LL_MPU_REGION_NUMBER7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MPU_RASR ENABLE LL_MPU_EnableRegion</li> </ul>                                                                                                                                                                                                                                                                                                                                                 |

**LL\_MPU\_ConfigRegion**

|                      |                                                                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_MPU_ConfigRegion (uint32_t Region, uint32_t SubRegionDisable, uint32_t Address, uint32_t Attributes)</code></b>                                                                                                                                                    |
| Function description | Configure and enable a region.                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Region:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_MPU_REGION_NUMBER0</li> <li>- LL_MPU_REGION_NUMBER1</li> <li>- LL_MPU_REGION_NUMBER2</li> <li>- LL_MPU_REGION_NUMBER3</li> </ul> </li> </ul> |

- LL\_MPUM\_REGION\_NUMBER4
- LL\_MPUM\_REGION\_NUMBER5
- LL\_MPUM\_REGION\_NUMBER6
- LL\_MPUM\_REGION\_NUMBER7
- **Address:** Value of region base address
- **SubRegionDisable:** Sub-region disable value between Min\_Data = 0x00 and Max\_Data = 0xFF
- **Attributes:** This parameter can be a combination of the following values:
  - LL\_MPUM\_REGION\_SIZE\_32B or  
LL\_MPUM\_REGION\_SIZE\_64B or  
LL\_MPUM\_REGION\_SIZE\_128B or  
LL\_MPUM\_REGION\_SIZE\_256B or  
LL\_MPUM\_REGION\_SIZE\_512B or  
LL\_MPUM\_REGION\_SIZE\_1KB or  
LL\_MPUM\_REGION\_SIZE\_2KB or  
LL\_MPUM\_REGION\_SIZE\_4KB or  
LL\_MPUM\_REGION\_SIZE\_8KB or  
LL\_MPUM\_REGION\_SIZE\_16KB or  
LL\_MPUM\_REGION\_SIZE\_32KB or  
LL\_MPUM\_REGION\_SIZE\_64KB or  
LL\_MPUM\_REGION\_SIZE\_128KB or  
LL\_MPUM\_REGION\_SIZE\_256KB or  
LL\_MPUM\_REGION\_SIZE\_512KB or  
LL\_MPUM\_REGION\_SIZE\_1MB or  
LL\_MPUM\_REGION\_SIZE\_2MB or  
LL\_MPUM\_REGION\_SIZE\_4MB or  
LL\_MPUM\_REGION\_SIZE\_8MB or  
LL\_MPUM\_REGION\_SIZE\_16MB or  
LL\_MPUM\_REGION\_SIZE\_32MB or  
LL\_MPUM\_REGION\_SIZE\_64MB or  
LL\_MPUM\_REGION\_SIZE\_128MB or  
LL\_MPUM\_REGION\_SIZE\_256MB or  
LL\_MPUM\_REGION\_SIZE\_512MB or  
LL\_MPUM\_REGION\_SIZE\_1GB or  
LL\_MPUM\_REGION\_SIZE\_2GB or  
LL\_MPUM\_REGION\_SIZE\_4GB
  - LL\_MPUM\_REGION\_NO\_ACCESS or  
LL\_MPUM\_REGION\_PRIV\_RW or  
LL\_MPUM\_REGION\_PRIV\_RW\_URO or  
LL\_MPUM\_REGION\_FULL\_ACCESS or  
LL\_MPUM\_REGION\_PRIV\_RO or  
LL\_MPUM\_REGION\_PRIV\_RO\_URO
  - LL\_MPUM\_TEX\_LEVEL0 or LL\_MPUM\_TEX\_LEVEL1 or  
LL\_MPUM\_TEX\_LEVEL2 or LL\_MPUM\_TEX\_LEVEL4
  - LL\_MPUM\_INSTRUCTION\_ACCESS\_ENABLE or  
LL\_MPUM\_INSTRUCTION\_ACCESS\_DISABLE
  - LL\_MPUM\_ACCESS\_SHAREABLE or  
LL\_MPUM\_ACCESS\_NOT\_SHAREABLE
  - LL\_MPUM\_ACCESS\_CACHEABLE or  
LL\_MPUM\_ACCESS\_NOT\_CACHEABLE
  - LL\_MPUM\_ACCESS\_BUFFERABLE or  
LL\_MPUM\_ACCESS\_NOT\_BUFFERABLE

---

**Return values**

- **None**

- Reference Manual to  
LL API cross  
reference:
- MPU\_RNR REGION LL\_MPU\_ConfigRegion
  - MPU\_RBAR REGION LL\_MPU\_ConfigRegion
  - MPU\_RBAR ADDR LL\_MPU\_ConfigRegion
  - MPU\_RASR XN LL\_MPU\_ConfigRegion
  - MPU\_RASR AP LL\_MPU\_ConfigRegion
  - MPU\_RASR S LL\_MPU\_ConfigRegion
  - MPU\_RASR C LL\_MPU\_ConfigRegion
  - MPU\_RASR B LL\_MPU\_ConfigRegion
  - MPU\_RASR SIZE LL\_MPU\_ConfigRegion

### **LL\_MPU\_DisableRegion**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_MPU_DisableRegion (uint32_t Region)</code></b>                                                                                                                                                                                                                                                                                                                                                                   |
| Function description                              | Disable a region.                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Region:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_MPU_REGION_NUMBER0</li> <li>- LL_MPU_REGION_NUMBER1</li> <li>- LL_MPU_REGION_NUMBER2</li> <li>- LL_MPU_REGION_NUMBER3</li> <li>- LL_MPU_REGION_NUMBER4</li> <li>- LL_MPU_REGION_NUMBER5</li> <li>- LL_MPU_REGION_NUMBER6</li> <li>- LL_MPU_REGION_NUMBER7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MPU_RNR REGION LL_MPU_DisableRegion</li> <li>• MPU_RASR ENABLE LL_MPU_DisableRegion</li> </ul>                                                                                                                                                                                                                                                                                                           |

## **62.2 CORTEX Firmware driver defines**

### **62.2.1 CORTEX**

#### ***MPU Bufferable Access***

`LL_MPU_ACCESS_BUFFERABLE`      Bufferable memory attribute

`LL_MPU_ACCESS_NOT_BUFFERABLE`    Not Bufferable memory attribute

#### ***MPU Cacheable Access***

`LL_MPU_ACCESS_CACHEABLE`        Cacheable memory attribute

`LL_MPU_ACCESS_NOT_CACHEABLE`    Not Cacheable memory attribute

#### ***SYSTICK Clock Source***

`LL_SYSTICK_CLKSOURCE_HCLK_DIV8`   AHB clock divided by 8 selected as SysTick clock source.

`LL_SYSTICK_CLKSOURCE_HCLK`        AHB clock selected as SysTick clock source.

#### ***MPU Control***

`LL_MPU_CTRL_HFNMI_PRIVDEF_NONE`   Disable NMI and privileged SW access

|                                   |                                                                             |
|-----------------------------------|-----------------------------------------------------------------------------|
| LL_MPU_CTRL_HARDFAULT_NMI         | Enables the operation of MPU during hard fault, NMI, and FAULTMASK handlers |
| LL_MPU_CTRL_PRIVILEGED_DEFAULT    | Enable privileged software access to default memory map                     |
| LL_MPU_CTRL_HFNMI_PRIVDEF         | Enable NMI and privileged SW access                                         |
| <b>Handler Fault type</b>         |                                                                             |
| LL_HANDLER_FAULT_USG              | Usage fault                                                                 |
| LL_HANDLER_FAULT_BUS              | Bus fault                                                                   |
| LL_HANDLER_FAULT_MEM              | Memory management fault                                                     |
| <b>MPU Instruction Access</b>     |                                                                             |
| LL_MPU_INSTRUCTION_ACCESS_ENABLE  | Instruction fetches enabled                                                 |
| LL_MPU_INSTRUCTION_ACCESS_DISABLE | Instruction fetches disabled                                                |
| <b>MPU Region Number</b>          |                                                                             |
| LL_MPU_REGION_NUMBER0             | REGION Number 0                                                             |
| LL_MPU_REGION_NUMBER1             | REGION Number 1                                                             |
| LL_MPU_REGION_NUMBER2             | REGION Number 2                                                             |
| LL_MPU_REGION_NUMBER3             | REGION Number 3                                                             |
| LL_MPU_REGION_NUMBER4             | REGION Number 4                                                             |
| LL_MPU_REGION_NUMBER5             | REGION Number 5                                                             |
| LL_MPU_REGION_NUMBER6             | REGION Number 6                                                             |
| LL_MPU_REGION_NUMBER7             | REGION Number 7                                                             |
| <b>MPU Region Privileges</b>      |                                                                             |
| LL_MPU_REGION_NO_ACCESS           | No access                                                                   |
| LL_MPU_REGION_PRIV_RW             | RW privileged (privileged access only)                                      |
| LL_MPU_REGION_PRIV_RW_URO         | RW privileged - RO user (Write in a user program generates a fault)         |
| LL_MPU_REGION_FULL_ACCESS         | RW privileged & user (Full access)                                          |
| LL_MPU_REGION_PRIV_RO             | RO privileged (privileged read only)                                        |
| LL_MPU_REGION_PRIV_RO_URO         | RO privileged & user (read only)                                            |
| <b>MPU Region Size</b>            |                                                                             |
| LL_MPU_REGION_SIZE_32B            | 32B Size of the MPU protection region                                       |
| LL_MPU_REGION_SIZE_64B            | 64B Size of the MPU protection region                                       |
| LL_MPU_REGION_SIZE_128B           | 128B Size of the MPU protection region                                      |
| LL_MPU_REGION_SIZE_256B           | 256B Size of the MPU protection region                                      |
| LL_MPU_REGION_SIZE_512B           | 512B Size of the MPU protection region                                      |
| LL_MPU_REGION_SIZE_1KB            | 1KB Size of the MPU protection region                                       |
| LL_MPU_REGION_SIZE_2KB            | 2KB Size of the MPU protection region                                       |

|                          |                                         |
|--------------------------|-----------------------------------------|
| LL_MPU_REGION_SIZE_4KB   | 4KB Size of the MPU protection region   |
| LL_MPU_REGION_SIZE_8KB   | 8KB Size of the MPU protection region   |
| LL_MPU_REGION_SIZE_16KB  | 16KB Size of the MPU protection region  |
| LL_MPU_REGION_SIZE_32KB  | 32KB Size of the MPU protection region  |
| LL_MPU_REGION_SIZE_64KB  | 64KB Size of the MPU protection region  |
| LL_MPU_REGION_SIZE_128KB | 128KB Size of the MPU protection region |
| LL_MPU_REGION_SIZE_256KB | 256KB Size of the MPU protection region |
| LL_MPU_REGION_SIZE_512KB | 512KB Size of the MPU protection region |
| LL_MPU_REGION_SIZE_1MB   | 1MB Size of the MPU protection region   |
| LL_MPU_REGION_SIZE_2MB   | 2MB Size of the MPU protection region   |
| LL_MPU_REGION_SIZE_4MB   | 4MB Size of the MPU protection region   |
| LL_MPU_REGION_SIZE_8MB   | 8MB Size of the MPU protection region   |
| LL_MPU_REGION_SIZE_16MB  | 16MB Size of the MPU protection region  |
| LL_MPU_REGION_SIZE_32MB  | 32MB Size of the MPU protection region  |
| LL_MPU_REGION_SIZE_64MB  | 64MB Size of the MPU protection region  |
| LL_MPU_REGION_SIZE_128MB | 128MB Size of the MPU protection region |
| LL_MPU_REGION_SIZE_256MB | 256MB Size of the MPU protection region |
| LL_MPU_REGION_SIZE_512MB | 512MB Size of the MPU protection region |
| LL_MPU_REGION_SIZE_1GB   | 1GB Size of the MPU protection region   |
| LL_MPU_REGION_SIZE_2GB   | 2GB Size of the MPU protection region   |
| LL_MPU_REGION_SIZE_4GB   | 4GB Size of the MPU protection region   |

***MPU Shareable Access***

|                             |                                |
|-----------------------------|--------------------------------|
| LL_MPU_ACCESS_SHAREABLE     | Shareable memory attribute     |
| LL_MPU_ACCESS_NOT_SHAREABLE | Not Shareable memory attribute |

***MPU TEX Level***

|                   |                   |
|-------------------|-------------------|
| LL_MPU_TEX_LEVEL0 | b000 for TEX bits |
| LL_MPU_TEX_LEVEL1 | b001 for TEX bits |
| LL_MPU_TEX_LEVEL2 | b010 for TEX bits |
| LL_MPU_TEX_LEVEL4 | b100 for TEX bits |

## 63 LL CRC Generic Driver

### 63.1 CRC Firmware driver API description

#### 63.1.1 Detailed description of functions

##### **LL\_CRC\_ResetCRCCalculationUnit**

Function name      **\_\_STATIC\_INLINE void LL\_CRC\_ResetCRCCalculationUnit(CRC\_TypeDef \* CRCx)**

Function description      Reset the CRC calculation unit.

Parameters      • **CRCx:** CRC Instance

Return values      • **None**

Notes      • If Programmable Initial CRC value feature is available, also set the Data Register to the value stored in the CRC\_INIT register, otherwise, reset Data Register to its default value.

Reference Manual to  
LL API cross  
reference:      • CR RESET LL\_CRC\_ResetCRCCalculationUnit

##### **LL\_CRC\_SetPolynomialSize**

Function name      **\_\_STATIC\_INLINE void LL\_CRC\_SetPolynomialSize(CRC\_TypeDef \* CRCx, uint32\_t PolySize)**

Function description      Configure size of the polynomial.

Parameters      • **CRCx:** CRC Instance

• **PolySize:** This parameter can be one of the following values:  
 – LL\_CRC\_POLYLENGTH\_32B  
 – LL\_CRC\_POLYLENGTH\_16B  
 – LL\_CRC\_POLYLENGTH\_8B  
 – LL\_CRC\_POLYLENGTH\_7B

Return values      • **None**

Reference Manual to  
LL API cross  
reference:      • CR POLYSIZE LL\_CRC\_SetPolynomialSize

##### **LL\_CRC\_GetPolynomialSize**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_CRC\_GetPolynomialSize(CRC\_TypeDef \* CRCx)**

Function description      Return size of the polynomial.

Parameters      • **CRCx:** CRC Instance

Return values      • **Returned:** value can be one of the following values:  
 – LL\_CRC\_POLYLENGTH\_32B  
 – LL\_CRC\_POLYLENGTH\_16B

- LL\_CRC\_POLYLENGTH\_8B
- LL\_CRC\_POLYLENGTH\_7B

Reference Manual to  
LL API cross  
reference:

- CR POLYSIZE LL\_CRC\_GetPolynomialSize

### **LL\_CRC\_SetInputDataReverseMode**

Function name      **\_\_STATIC\_INLINE void LL\_CRC\_SetInputDataReverseMode (CRC\_TypeDef \* CRCx, uint32\_t ReverseMode)**

Function description      Configure the reversal of the bit order of the input data.

- Parameters
- **CRCx:** CRC Instance
  - **ReverseMode:** This parameter can be one of the following values:
    - LL\_CRC\_INDATA\_REVERSE\_NONE
    - LL\_CRC\_INDATA\_REVERSE\_BYTE
    - LL\_CRC\_INDATA\_REVERSE\_HALFWORD
    - LL\_CRC\_INDATA\_REVERSE\_WORD

- Return values
- **None**

Reference Manual to  
LL API cross  
reference:

- CR REV\_IN LL\_CRC\_SetInputDataReverseMode

### **LL\_CRC\_GetInputDataReverseMode**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_CRC\_GetInputDataReverseMode (CRC\_TypeDef \* CRCx)**

Function description      Return type of reversal for input data bit order.

- Parameters
- **CRCx:** CRC Instance

- Return values
- **Returned:** value can be one of the following values:
    - LL\_CRC\_INDATA\_REVERSE\_NONE
    - LL\_CRC\_INDATA\_REVERSE\_BYTE
    - LL\_CRC\_INDATA\_REVERSE\_HALFWORD
    - LL\_CRC\_INDATA\_REVERSE\_WORD

Reference Manual to  
LL API cross  
reference:

- CR REV\_IN LL\_CRC\_GetInputDataReverseMode

### **LL\_CRC\_SetOutputDataReverseMode**

Function name      **\_\_STATIC\_INLINE void LL\_CRC\_SetOutputDataReverseMode (CRC\_TypeDef \* CRCx, uint32\_t ReverseMode)**

Function description      Configure the reversal of the bit order of the Output data.

- Parameters
- **CRCx:** CRC Instance
  - **ReverseMode:** This parameter can be one of the following values:
    - LL\_CRC\_OUTDATA\_REVERSE\_NONE
    - LL\_CRC\_OUTDATA\_REVERSE\_BIT

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR REV_OUT LL_CRC_SetOutputDataReverseMode</li> </ul> |

### LL\_CRC\_GetOutputDataReverseMode

|                                                   |                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_CRC_GetOutputDataReverseMode (CRC_TypeDef * CRCx)</code>                                                                                                                                            |
| Function description                              | Configure the reversal of the bit order of the Output data.                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>CRCx:</b> CRC Instance</li> </ul>                                                                                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>LL_CRC_OUTDATA_REVERSE_NONE</li> <li>LL_CRC_OUTDATA_REVERSE_BIT</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR REV_OUT LL_CRC_GetOutputDataReverseMode</li> </ul>                                                                                                                                              |

### LL\_CRC\_SetInitialData

|                                                   |                                                                                                                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_CRC_SetInitialData (CRC_TypeDef<br/>* CRCx, uint32_t InitCrc)</code>                                                                                                                                         |
| Function description                              | Initialize the Programmable initial CRC value.                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>CRCx:</b> CRC Instance</li> <li><b>InitCrc:</b> Value to be programmed in Programmable initial CRC value register</li> </ul>                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>If the CRC size is less than 32 bits, the least significant bits are used to write the correct value</li> <li>LL_CRC_DEFAULT_CRC_INITVALUE could be used as value for InitCrc parameter.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>INIT INIT LL_CRC_SetInitialData</li> </ul>                                                                                                                                                          |

### LL\_CRC\_GetInitialData

|                      |                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_CRC_GetInitialData<br/>(CRC_TypeDef * CRCx)</code>                                                                  |
| Function description | Return current Initial CRC value.                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>CRCx:</b> CRC Instance</li> </ul>                                                                           |
| Return values        | <ul style="list-style-type: none"> <li><b>Value:</b> programmed in Programmable initial CRC value register</li> </ul>                                 |
| Notes                | <ul style="list-style-type: none"> <li>If the CRC size is less than 32 bits, the least significant bits are used to read the correct value</li> </ul> |

- Reference Manual to LL API cross reference:
- INIT INIT LL\_CRC\_GetInitialData

### **LL\_CRC\_SetPolynomialCoef**

|                                             |                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_CRC_SetPolynomialCoef (CRC_TypeDef * CRCx, uint32_t PolynomCoef)</code></b>                                                                                                                                                                                                                                                    |
| Function description                        | Initialize the Programmable polynomial value (coefficients of the polynomial to be used for CRC calculation).                                                                                                                                                                                                                                                  |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>CRCx:</b> CRC Instance</li> <li><b>PolynomCoef:</b> Value to be programmed in Programmable Polynomial value register</li> </ul>                                                                                                                                                                                      |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                  |
| Notes                                       | <ul style="list-style-type: none"> <li>LL_CRC_DEFAULT_CRC32_POLY could be used as value for PolynomCoef parameter.</li> <li>Please check Reference Manual and existing Errata Sheets, regarding possible limitations for Polynomial values usage. For example, for a polynomial of degree 7, <math>X^7 + X^6 + X^5 + X^2 + 1</math> is written 0x65</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>POL POL LL_CRC_SetPolynomialCoef</li> </ul>                                                                                                                                                                                                                                                                             |

### **LL\_CRC\_GetPolynomialCoef**

|                                             |                                                                                                                                                                                                                                                                           |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_CRC_GetPolynomialCoef (CRC_TypeDef * CRCx)</code></b>                                                                                                                                                                                 |
| Function description                        | Return current Programmable polynomial value.                                                                                                                                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>CRCx:</b> CRC Instance</li> </ul>                                                                                                                                                                                               |
| Return values                               | <ul style="list-style-type: none"> <li><b>Value:</b> programmed in Programmable Polynomial value register</li> </ul>                                                                                                                                                      |
| Notes                                       | <ul style="list-style-type: none"> <li>Please check Reference Manual and existing Errata Sheets, regarding possible limitations for Polynomial values usage. For example, for a polynomial of degree 7, <math>X^7 + X^6 + X^5 + X^2 + 1</math> is written 0x65</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>POL POL LL_CRC_GetPolynomialCoef</li> </ul>                                                                                                                                                                                        |

### **LL\_CRC\_FeedData32**

|                      |                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_CRC_FeedData32 (CRC_TypeDef * CRCx, uint32_t InData)</code></b>                                                    |
| Function description | Write given 32-bit data to the CRC calculator.                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>CRCx:</b> CRC Instance</li> <li><b>InData:</b> value to be provided to CRC calculator between</li> </ul> |

between Min\_Data=0 and Max\_Data=0xFFFFFFFF

Return values

- **None**
- DR DR LL\_CRC\_FeedData32

Reference Manual to  
LL API cross  
reference:

### **LL\_CRC\_FeedData16**

Function name

`__STATIC_INLINE void LL_CRC_FeedData16 (CRC_TypeDef *  
CRCx, uint16_t InData)`

Function description

Write given 16-bit data to the CRC calculator.

Parameters

- **CRCx:** CRC Instance
- **InData:** 16 bit value to be provided to CRC calculator  
between Min\_Data=0 and Max\_Data=0xFFFF

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- DR DR LL\_CRC\_FeedData16

### **LL\_CRC\_FeedData8**

Function name

`__STATIC_INLINE void LL_CRC_FeedData8 (CRC_TypeDef *  
CRCx, uint8_t InData)`

Function description

Write given 8-bit data to the CRC calculator.

Parameters

- **CRCx:** CRC Instance
- **InData:** 8 bit value to be provided to CRC calculator between  
Min\_Data=0 and Max\_Data=0xFF

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- DR DR LL\_CRC\_FeedData8

### **LL\_CRC\_ReadData32**

Function name

`__STATIC_INLINE uint32_t LL_CRC_ReadData32  
(CRC_TypeDef * CRCx)`

Function description

Return current CRC calculation result.

Parameters

- **CRCx:** CRC Instance

Return values

- **Current:** CRC calculation result as stored in CRC\_DR  
register (32 bits).

Reference Manual to  
LL API cross  
reference:

- DR DR LL\_CRC\_ReadData32

**LL\_CRC\_ReadData16**

|                                                   |                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint16_t LL_CRC_ReadData16(CRC_TypeDef * CRCx)</code>                                                     |
| Function description                              | Return current CRC calculation result.                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"><li><b>CRCx:</b> CRC Instance</li></ul>                                                      |
| Return values                                     | <ul style="list-style-type: none"><li><b>Current:</b> CRC calculation result as stored in CRC_DR register (16 bits).</li></ul> |
| Notes                                             | <ul style="list-style-type: none"><li>This function is expected to be used in a 16 bits CRC polynomial size context.</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>DR DR LL_CRC_ReadData16</li></ul>                                                        |

**LL\_CRC\_ReadData8**

|                                                   |                                                                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint8_t LL_CRC_ReadData8(CRC_TypeDef * CRCx)</code>                                                      |
| Function description                              | Return current CRC calculation result.                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"><li><b>CRCx:</b> CRC Instance</li></ul>                                                     |
| Return values                                     | <ul style="list-style-type: none"><li><b>Current:</b> CRC calculation result as stored in CRC_DR register (8 bits).</li></ul> |
| Notes                                             | <ul style="list-style-type: none"><li>This function is expected to be used in a 8 bits CRC polynomial size context.</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>DR DR LL_CRC_ReadData8</li></ul>                                                        |

**LL\_CRC\_ReadData7**

|                                                   |                                                                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint8_t LL_CRC_ReadData7(CRC_TypeDef * CRCx)</code>                                                      |
| Function description                              | Return current CRC calculation result.                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"><li><b>CRCx:</b> CRC Instance</li></ul>                                                     |
| Return values                                     | <ul style="list-style-type: none"><li><b>Current:</b> CRC calculation result as stored in CRC_DR register (7 bits).</li></ul> |
| Notes                                             | <ul style="list-style-type: none"><li>This function is expected to be used in a 7 bits CRC polynomial size context.</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>DR DR LL_CRC_ReadData7</li></ul>                                                        |

**LL\_CRC\_Read\_IDR**

|                                                   |                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_CRC_Read_IDR (CRC_TypeDef * CRCx)</code>                                                           |
| Function description                              | Return data stored in the Independent Data(IDR) register.                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>CRCx:</b> CRC Instance</li> </ul>                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> stored in CRC_IDR register (General-purpose 8-bit data register).</li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• This register can be used as a temporary storage location for one byte.</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IDR IDR LL_CRC_Read_IDR</li> </ul>                                                         |

**LL\_CRC\_Write\_IDR**

|                                                   |                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_CRC_Write_IDR (CRC_TypeDef * CRCx, uint32_t InData)</code>                                                                                                     |
| Function description                              | Store data in the Independent Data(IDR) register.                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>CRCx:</b> CRC Instance</li> <li>• <b>InData:</b> value to be stored in CRC_IDR register (8-bit) between Min_Data=0 and Max_Data=0xFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• This register can be used as a temporary storage location for one byte.</li> </ul>                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IDR IDR LL_CRC_Write_IDR</li> </ul>                                                                                                                |

**LL\_CRC\_Delnit**

|                      |                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_CRC_Delnit (CRC_TypeDef * CRCx)</code>                                                                                                                                                                                         |
| Function description | De-initialize CRC registers (Registers restored to their default values).                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>CRCx:</b> CRC Instance</li> </ul>                                                                                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: CRC registers are de-initialized</li> <li>– ERROR: CRC registers are not de-initialized</li> </ul> </li> </ul> |

## 63.2 CRC Firmware driver defines

### 63.2.1 CRC

*Default CRC computation initialization value*

`LL_CRC_DEFAULT_CRC_INITVALUE` Default CRC computation initialization value

***Default CRC generating polynomial value***

`LL_CRC_DEFAULT_CRC32_POLY` Default CRC generating polynomial value

***Input Data Reverse***

|                                             |                                           |
|---------------------------------------------|-------------------------------------------|
| <code>LL_CRC_INDATA_REVERSE_NONE</code>     | Input Data bit order not affected         |
| <code>LL_CRC_INDATA_REVERSE_BYTE</code>     | Input Data bit reversal done by byte      |
| <code>LL_CRC_INDATA_REVERSE_HALFWORD</code> | Input Data bit reversal done by half-word |
| <code>LL_CRC_INDATA_REVERSE_WORD</code>     | Input Data bit reversal done by word      |

***Output Data Reverse***

|                                          |                                      |
|------------------------------------------|--------------------------------------|
| <code>LL_CRC_OUTDATA_REVERSE_NONE</code> | Output Data bit order not affected   |
| <code>LL_CRC_OUTDATA_REVERSE_BIT</code>  | Output Data bit reversal done by bit |

***Polynomial length***

|                                    |                         |
|------------------------------------|-------------------------|
| <code>LL_CRC_POLYLENGTH_32B</code> | 32 bits Polynomial size |
| <code>LL_CRC_POLYLENGTH_16B</code> | 16 bits Polynomial size |
| <code>LL_CRC_POLYLENGTH_8B</code>  | 8 bits Polynomial size  |
| <code>LL_CRC_POLYLENGTH_7B</code>  | 7 bits Polynomial size  |

***Common Write and read registers Macros***

`LL_CRC_WriteReg`      **Description:**

- Write a value in CRC register.

**Parameters:**

- `_INSTANCE_`: CRC Instance
- `_REG_`: Register to be written
- `_VALUE_`: Value to be written in the register

**Return value:**

- None

`LL_CRC_ReadReg`      **Description:**

- Read a value in CRC register.

**Parameters:**

- `_INSTANCE_`: CRC Instance
- `_REG_`: Register to be read

**Return value:**

- Register: value

## 64 LL DAC Generic Driver

### 64.1 DAC Firmware driver registers structures

#### 64.1.1 LL\_DAC\_InitTypeDef

##### Data Fields

- *uint32\_t TriggerSource*
- *uint32\_t WaveAutoGeneration*
- *uint32\_t WaveAutoGenerationConfig*
- *uint32\_t OutputBuffer*

##### Field Documentation

- ***uint32\_t LL\_DAC\_InitTypeDef::TriggerSource***  
Set the conversion trigger source for the selected DAC channel: internal (SW start) or from external IP (timer event, external interrupt line). This parameter can be a value of **DAC\_LL\_EC\_TRIGGER\_SOURCE** This feature can be modified afterwards using unitary function **LL\_DAC\_SetTriggerSource()**.
- ***uint32\_t LL\_DAC\_InitTypeDef::WaveAutoGeneration***  
Set the waveform automatic generation mode for the selected DAC channel. This parameter can be a value of **DAC\_LL\_EC\_WAVE\_AUTO\_GENERATION\_MODE** This feature can be modified afterwards using unitary function **LL\_DAC\_SetWaveAutoGeneration()**.
- ***uint32\_t LL\_DAC\_InitTypeDef::WaveAutoGenerationConfig***  
Set the waveform automatic generation mode for the selected DAC channel. If waveform automatic generation mode is set to noise, this parameter can be a value of **DAC\_LL\_EC\_WAVE\_NOISE\_LFSR\_UNMASK\_BITS** If waveform automatic generation mode is set to triangle, this parameter can be a value of **DAC\_LL\_EC\_WAVE\_TRIANGLE\_AMPLITUDE**  
**Note:**If waveform automatic generation mode is disabled, this parameter is discarded. This feature can be modified afterwards using unitary function **LL\_DAC\_SetWaveNoiseLFSR()** or **LL\_DAC\_SetWaveTriangleAmplitude()**, depending on the wave automatic generation selected.
- ***uint32\_t LL\_DAC\_InitTypeDef::OutputBuffer***  
Set the output buffer for the selected DAC channel. This parameter can be a value of **DAC\_LL\_EC\_OUTPUT\_BUFFER** This feature can be modified afterwards using unitary function **LL\_DAC\_SetOutputBuffer()**.

### 64.2 DAC Firmware driver API description

#### 64.2.1 Detailed description of functions

##### LL\_DAC\_SetTriggerSource

Function name      **\_STATIC\_INLINE void LL\_DAC\_SetTriggerSource  
(DAC\_TypeDef \* DACx, uint32\_t DAC\_Channel, uint32\_t  
TriggerSource)**

Function description      Set the conversion trigger source for the selected DAC channel.

Parameters     
 

- **DACx:** DAC instance
- **DAC\_Channel:** This parameter can be one of the following

values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.

- LL\_DAC\_CHANNEL\_1
- LL\_DAC\_CHANNEL\_2 (1)

- **TriggerSource:** This parameter can be one of the following values: (1) On STM32F3, parameter not available on all devices (2) On STM32F3, parameter not available on all DAC instances: DAC1 (for DAC instances DACx available on the selected device).
  - LL\_DAC\_TRIG\_SOFTWARE
  - LL\_DAC\_TRIG\_EXT\_TIM2\_TRGO
  - LL\_DAC\_TRIG\_EXT\_TIM3\_TRGO (1)
  - LL\_DAC\_TRIG\_EXT\_TIM4\_TRGO (1)
  - LL\_DAC\_TRIG\_EXT\_TIM5\_TRGO (1)
  - LL\_DAC\_TRIG\_EXT\_TIM6\_TRGO
  - LL\_DAC\_TRIG\_EXT\_TIM7\_TRGO (1)
  - LL\_DAC\_TRIG\_EXT\_TIM8\_TRGO (1)
  - LL\_DAC\_TRIG\_EXT\_TIM15\_TRGO (1)
  - LL\_DAC\_TRIG\_EXT\_TIM18\_TRGO (1)
  - LL\_DAC\_TRIG\_EXT\_HRTIM1\_DACTRG1 (1)
  - LL\_DAC\_TRIG\_EXT\_HRTIM1\_DACTRG2 (1)(2)
  - LL\_DAC\_TRIG\_EXT\_HRTIM1\_DACTRG3 (1) (3)
  - LL\_DAC\_TRIG\_EXT EXTI\_LINE9
- (3) On STM32F3, parameter not available on all DAC instances: DAC2 (for DAC instances DACx available on the selected device).

#### Return values

- **None**

#### Notes

- For conversion trigger source to be effective, DAC trigger must be enabled using function `LL_DAC_EnableTrigger()`.
- To set conversion trigger source, DAC channel must be disabled. Otherwise, the setting is discarded.
- Availability of parameters of trigger sources from timer depends on timers availability on the selected device.

#### Reference Manual to LL API cross reference:

- CR TSEL1 `LL_DAC_SetTriggerSource`
- CR TSEL2 `LL_DAC_SetTriggerSource`

## **LL\_DAC\_GetTriggerSource**

#### Function name

```
__STATIC_INLINE uint32_t LL_DAC_GetTriggerSource
(DAC_TypeDef * DACx, uint32_t DAC_Channel)
```

#### Function description

Get the conversion trigger source for the selected DAC channel.

#### Parameters

- **DACx:** DAC instance
- **DAC\_Channel:** This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.
  - LL\_DAC\_CHANNEL\_1
  - LL\_DAC\_CHANNEL\_2 (1)

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (1) On STM32F3, parameter not available on all devices (2) On STM32F3, parameter not available on all DAC instances: DAC1 (for DAC instances DACx available on the selected device).           <ul style="list-style-type: none"> <li>– LL_DAC_TRIG_SOFTWARE</li> <li>– LL_DAC_TRIG_EXT_TIM2_TRGO</li> <li>– LL_DAC_TRIG_EXT_TIM3_TRGO (1)</li> <li>– LL_DAC_TRIG_EXT_TIM4_TRGO (1)</li> <li>– LL_DAC_TRIG_EXT_TIM5_TRGO (1)</li> <li>– LL_DAC_TRIG_EXT_TIM6_TRGO</li> <li>– LL_DAC_TRIG_EXT_TIM7_TRGO (1)</li> <li>– LL_DAC_TRIG_EXT_TIM8_TRGO (1)</li> <li>– LL_DAC_TRIG_EXT_TIM15_TRGO (1)</li> <li>– LL_DAC_TRIG_EXT_TIM18_TRGO (1)</li> <li>– LL_DAC_TRIG_EXT_HRTIM1_DACTRG1 (1)</li> <li>– LL_DAC_TRIG_EXT_HRTIM1_DACTRG2 (1)(2)</li> <li>– LL_DAC_TRIG_EXT_HRTIM1_DACTRG3 (1) (3)</li> <li>– LL_DAC_TRIG_EXT EXTI_LINE9</li> </ul> </li> <li>• (3) On STM32F3, parameter not available on all DAC instances: DAC2 (for DAC instances DACx available on the selected device).</li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• For conversion trigger source to be effective, DAC trigger must be enabled using function LL_DAC_EnableTrigger().</li> <li>• Availability of parameters of trigger sources from timer depends on timers availability on the selected device.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR TSEL1 LL_DAC_GetTriggerSource</li> <li>• CR TSEL2 LL_DAC_GetTriggerSource</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

## LL\_DAC\_SetWaveAutoGeneration

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DAC_SetWaveAutoGeneration(DAC_TypeDef * DACx, uint32_t DAC_Channel, uint32_t WaveAutoGeneration)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Set the waveform automatic generation mode for the selected DAC channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.           <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> <li>• <b>WaveAutoGeneration:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_DAC_WAVE_AUTO_GENERATION_NONE</li> <li>– LL_DAC_WAVE_AUTO_GENERATION_NOISE</li> <li>– LL_DAC_WAVE_AUTO_GENERATION_TRIANGLE</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

Reference Manual to  
LL API cross  
reference:

- CR WAVE1 LL\_DAC\_SetWaveAutoGeneration
- CR WAVE2 LL\_DAC\_SetWaveAutoGeneration

### **LL\_DAC\_GetWaveAutoGeneration**

Function name

**\_STATIC\_INLINE uint32\_t LL\_DAC\_GetWaveAutoGeneration  
(DAC\_TypeDef \* DACx, uint32\_t DAC\_Channel)**

Function description

Get the waveform automatic generation mode for the selected DAC channel.

Parameters

- **DACx:** DAC instance
- **DAC\_Channel:** This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.
  - LL\_DAC\_CHANNEL\_1
  - LL\_DAC\_CHANNEL\_2 (1)

Return values

- **Returned:** value can be one of the following values:
  - LL\_DAC\_WAVE\_AUTO\_GENERATION\_NONE
  - LL\_DAC\_WAVE\_AUTO\_GENERATION\_NOISE
  - LL\_DAC\_WAVE\_AUTO\_GENERATION\_TRIANGLE

Reference Manual to  
LL API cross  
reference:

- CR WAVE1 LL\_DAC\_SetWaveAutoGeneration
- CR WAVE2 LL\_DAC\_SetWaveAutoGeneration

### **LL\_DAC\_SetWaveNoiseLFSR**

Function name

**\_STATIC\_INLINE void LL\_DAC\_SetWaveNoiseLFSR  
(DAC\_TypeDef \* DACx, uint32\_t DAC\_Channel, uint32\_t  
NoiseLFSRMask)**

Function description

Set the noise waveform generation for the selected DAC channel:  
Noise mode and parameters LFSR (linear feedback shift register).

Parameters

- **DACx:** DAC instance
- **DAC\_Channel:** This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.
  - LL\_DAC\_CHANNEL\_1
  - LL\_DAC\_CHANNEL\_2 (1)
- **NoiseLFSRMask:** This parameter can be one of the following values:
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BIT0
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BITS1\_0
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BITS2\_0
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BITS3\_0
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BITS4\_0
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BITS5\_0
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BITS6\_0
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BITS7\_0
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BITS8\_0
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BITS9\_0
  - LL\_DAC\_NOISE\_LFSR\_UNMASK\_BITS10\_0

---

|                                                   |                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | – LL_DAC_NOISE_LFSR_UNMASK_BITS11_0                                                                                                                                                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• For wave generation to be effective, DAC channel wave generation mode must be enabled using function LL_DAC_SetWaveAutoGeneration().</li> <li>• This setting can be set when the selected DAC channel is disabled (otherwise, the setting operation is ignored).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR MAMP1 LL_DAC_SetWaveNoiseLFSR</li> <li>• CR MAMP2 LL_DAC_SetWaveNoiseLFSR</li> </ul>                                                                                                                                                                                     |

### LL\_DAC\_GetWaveNoiseLFSR

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><u>__STATIC_INLINE uint32_t LL_DAC_GetWaveNoiseLFSR(DAC_TypeDef * DACx, uint32_t DAC_Channel)</u></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Set the noise waveform generation for the selected DAC channel:<br>Noise mode and parameters LFSR (linear feedback shift register).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.             <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_DAC_NOISE_LFSR_UNMASK_BIT0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS1_0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS2_0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS3_0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS4_0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS5_0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS6_0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS7_0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS8_0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS9_0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS10_0</li> <li>– LL_DAC_NOISE_LFSR_UNMASK_BITS11_0</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR MAMP1 LL_DAC_SetWaveNoiseLFSR</li> <li>• CR MAMP2 LL_DAC_SetWaveNoiseLFSR</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

### LL\_DAC\_SetWaveTriangleAmplitude

|                      |                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><u>__STATIC_INLINE void LL_DAC_SetWaveTriangleAmplitude(DAC_TypeDef * DACx, uint32_t DAC_Channel, uint32_t TriangleAmplitude)</u></b> |
| Function description | Set the triangle waveform generation for the selected DAC channel: triangle mode and amplitude.                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> </ul>                                                            |

- **DAC\_Channel:** This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.
    - LL\_DAC\_CHANNEL\_1
    - LL\_DAC\_CHANNEL\_2 (1)
  - **TriangleAmplitude:** This parameter can be one of the following values:
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_1
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_3
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_7
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_15
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_31
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_63
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_127
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_255
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_511
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_1023
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_2047
    - LL\_DAC\_TRIANGLE\_AMPLITUDE\_4095
- Return values**
- **None**
- Notes**
- For wave generation to be effective, DAC channel wave generation mode must be enabled using function `LL_DAC_SetWaveAutoGeneration()`.
  - This setting can be set when the selected DAC channel is disabled (otherwise, the setting operation is ignored).
- Reference Manual to LL API cross reference:**
- CR MAMP1 `LL_DAC_SetWaveTriangleAmplitude`
  - CR MAMP2 `LL_DAC_SetWaveTriangleAmplitude`

### LL\_DAC\_GetWaveTriangleAmplitude

|                             |                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>        | <code>_STATIC_INLINE uint32_t<br/>LL_DAC_GetWaveTriangleAmplitude (DAC_TypeDef * DACx,<br/>uint32_t DAC_Channel)</code>                                                                                                                                                                                                                                                                               |
| <b>Function description</b> | Set the triangle waveform generation for the selected DAC channel: triangle mode and amplitude.                                                                                                                                                                                                                                                                                                       |
| <b>Parameters</b>           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.           <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> </ul> |
| <b>Return values</b>        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_DAC_TRIANGLE_AMPLITUDE_1</li> <li>– LL_DAC_TRIANGLE_AMPLITUDE_3</li> <li>– LL_DAC_TRIANGLE_AMPLITUDE_7</li> <li>– LL_DAC_TRIANGLE_AMPLITUDE_15</li> <li>– LL_DAC_TRIANGLE_AMPLITUDE_31</li> </ul> </li> </ul>                               |

- LL\_DAC\_TRIANGLE\_AMPLITUDE\_63
- LL\_DAC\_TRIANGLE\_AMPLITUDE\_127
- LL\_DAC\_TRIANGLE\_AMPLITUDE\_255
- LL\_DAC\_TRIANGLE\_AMPLITUDE\_511
- LL\_DAC\_TRIANGLE\_AMPLITUDE\_1023
- LL\_DAC\_TRIANGLE\_AMPLITUDE\_2047
- LL\_DAC\_TRIANGLE\_AMPLITUDE\_4095

Reference Manual to  
LL API cross  
reference:

- CR MAMP1 LL\_DAC\_GetWaveTriangleAmplitude
- CR MAMP2 LL\_DAC\_GetWaveTriangleAmplitude

### **LL\_DAC\_SetOutputBuffer**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DAC_SetOutputBuffer(DAC_TypeDef * DACx, uint32_t DAC_Channel, uint32_t OutputBuffer)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description                              | Set the output buffer for the selected DAC channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>- LL_DAC_CHANNEL_1</li> <li>- LL_DAC_CHANNEL_2 (1)</li> </ul> </li> <li>• <b>OutputBuffer:</b> This parameter can be one of the following values: (1) Feature specific to STM32F303x6/8 and STM32F328: On DAC1 channel 2, output buffer is replaced by a switch to connect DAC channel output to pin PA5. On DAC2 channel 1, output buffer is replaced by a switch to connect DAC channel output to pin PA6. <ul style="list-style-type: none"> <li>- LL_DAC_OUTPUT_BUFFER_ENABLE</li> <li>- LL_DAC_OUTPUT_BUFFER_DISABLE</li> <li>- LL_DAC_OUTPUT_SWITCH_DISABLE (1)</li> <li>- LL_DAC_OUTPUT_SWITCH_ENABLE (1)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR BOFF1 LL_DAC_SetOutputBuffer</li> <li>• CR BOFF2 LL_DAC_SetOutputBuffer</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

### **LL\_DAC\_GetOutputBuffer**

|                      |                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DAC_GetOutputBuffer(DAC_TypeDef * DACx, uint32_t DAC_Channel)</code>                                                                                                                                                                                                                                                      |
| Function description | Get the output buffer state for the selected DAC channel.                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>- LL_DAC_CHANNEL_1</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | – LL_DAC_CHANNEL_2 (1)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (1) Feature specific to STM32F303x6/8 and STM32F328: On DAC1 channel 2, output buffer is replaced by a switch to connect DAC channel output to pin PA5. On DAC2 channel 1, output buffer is replaced by a switch to connect DAC channel output to pin PA6.           <ul style="list-style-type: none"> <li>– LL_DAC_OUTPUT_BUFFER_ENABLE</li> <li>– LL_DAC_OUTPUT_BUFFER_DISABLE</li> <li>– LL_DAC_OUTPUT_SWITCH_DISABLE (1)</li> <li>– LL_DAC_OUTPUT_SWITCH_ENABLE (1)</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR BOFF1 LL_DAC_GetOutputBuffer</li> <li>• CR BOFF2 LL_DAC_GetOutputBuffer</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

### **LL\_DAC\_EnableDMAReq**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><u>__STATIC_INLINE void LL_DAC_EnableDMAReq(DAC_TypeDef * DACx, uint32_t DAC_Channel)</u></b>                                                                                                                                                                                                                                                                                                      |
| Function description                              | Enable DAC DMA transfer request of the selected channel.                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.           <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• To configure DMA source address (peripheral address), use function LL_DAC_DMA_GetRegAddr().</li> </ul>                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR DMAEN1 LL_DAC_EnableDMAReq</li> <li>• CR DMAEN2 LL_DAC_EnableDMAReq</li> </ul>                                                                                                                                                                                                                                                                            |

### **LL\_DAC\_DisableDMAReq**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><u>__STATIC_INLINE void LL_DAC_DisableDMAReq(DAC_TypeDef * DACx, uint32_t DAC_Channel)</u></b>                                                                                                                                                                                                                                                                                                     |
| Function description | Disable DAC DMA transfer request of the selected channel.                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.           <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• To configure DMA source address (peripheral address), use</li> </ul>                                                                                                                                                                                                                                                                                         |

---

|                                                   |                                                                                                                              |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR DMAEN1 LL_DAC_DisableDMAReq</li> <li>• CR DMAEN2 LL_DAC_DisableDMAReq</li> </ul> |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|

### LL\_DAC\_IsDMAReqEnabled

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DAC_IsDMAReqEnabled(DAC_TypeDef * DACx, uint32_t DAC_Channel)</code>                                                                                                                                                                                                                                                                                      |
| Function description                              | Get DAC DMA transfer request state of the selected channel.                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR DMAEN1 LL_DAC_IsDMAReqEnabled</li> <li>• CR DMAEN2 LL_DAC_IsDMAReqEnabled</li> </ul>                                                                                                                                                                                                                                                            |

### LL\_DAC\_DMA\_GetRegAddr

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DAC_DMA_GetRegAddr(DAC_TypeDef * DACx, uint32_t DAC_Channel, uint32_t Register)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Function to help to configure DMA transfer to DAC: retrieve the DAC register address from DAC instance and a list of DAC registers intended to be used (most commonly) with DMA transfer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> <li>• <b>Register:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DAC_DMA_REG_DATA_12BITS_RIGHT_ALIGNED</li> <li>– LL_DAC_DMA_REG_DATA_12BITS_LEFT_ALIGNED</li> <li>– LL_DAC_DMA_REG_DATA_8BITS_RIGHT_ALIGNED</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>DAC:</b> register address</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>• These DAC registers are data holding registers: when DAC conversion is requested, DAC generates a DMA transfer request to have data available in DAC data holding registers.</li> <li>• This macro is intended to be used with LL DMA driver, refer to function "LL_DMA_ConfigAddresses()". Example:<br/> <code>LL_DMA_ConfigAddresses(DMA1, LL_DMA_CHANNEL_1, (uint32_t)&amp;&lt; array or variable &gt;, LL_DAC_DMA_GetRegAddr(DAC1, LL_DAC_CHANNEL_1, LL_DAC_DMA_REG_DATA_12BITS_RIGHT_ALIGNED),</code> </li> </ul>                                                                                                                |

- Reference Manual to LL API cross reference:
- DHR12R1 DACC1DHR LL\_DAC\_DMA\_GetRegAddr
  - DHR12L1 DACC1DHR LL\_DAC\_DMA\_GetRegAddr
  - DHR8R1 DACC1DHR LL\_DAC\_DMA\_GetRegAddr
  - DHR12R2 DACC2DHR LL\_DAC\_DMA\_GetRegAddr
  - DHR12L2 DACC2DHR LL\_DAC\_DMA\_GetRegAddr
  - DHR8R2 DACC2DHR LL\_DAC\_DMA\_GetRegAddr

### **LL\_DAC\_Enable**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_DAC_Enable (DAC_TypeDef * DACx, uint32_t DAC_Channel)</code></b>                                                                                                                                                                                                                                                                                                                      |
| Function description                        | Enable DAC selected channel.                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>- <code>LL_DAC_CHANNEL_1</code></li> <li>- <code>LL_DAC_CHANNEL_2</code> (1)</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                       |
| Notes                                       | <ul style="list-style-type: none"> <li>• After enable from off state, DAC channel requires a delay for output voltage to reach accuracy +/- 1 LSB. Refer to device datasheet, parameter "tWAKEUP".</li> </ul>                                                                                                                                                                                                         |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR EN1 <code>LL_DAC_Enable</code></li> <li>• CR EN2 <code>LL_DAC_Enable</code></li> </ul>                                                                                                                                                                                                                                                                                    |

### **LL\_DAC\_Disable**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_DAC_Disable (DAC_TypeDef * DACx, uint32_t DAC_Channel)</code></b>                                                                                                                                                                                                                                                                                                                     |
| Function description                        | Disable DAC selected channel.                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>- <code>LL_DAC_CHANNEL_1</code></li> <li>- <code>LL_DAC_CHANNEL_2</code> (1)</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR EN1 <code>LL_DAC_Disable</code></li> <li>• CR EN2 <code>LL_DAC_Disable</code></li> </ul>                                                                                                                                                                                                                                                                                  |

**LL\_DAC\_IsEnabled**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_DAC_IsEnabled (DAC_TypeDef * DACx, uint32_t DAC_Channel)</code>                                                                                                                                                                                                                                                                                           |
| Function description                        | Get DAC enable state of the selected channel.                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                          |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR EN1 LL_DAC_IsEnabled</li> <li>• CR EN2 LL_DAC_IsEnabled</li> </ul>                                                                                                                                                                                                                                                                              |

**LL\_DAC\_EnableTrigger**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_DAC_EnableTrigger (DAC_TypeDef * DACx, uint32_t DAC_Channel)</code>                                                                                                                                                                                                                                                                                                                                                                     |
| Function description                        | Enable DAC trigger of the selected channel.                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> </ul>                                                                           |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                                       | <ul style="list-style-type: none"> <li>• - If DAC trigger is disabled, DAC conversion is performed automatically once the data holding register is updated, using functions "LL_DAC_ConvertData{8; 12}{Right; Left}Aligned()": LL_DAC_ConvertData12RightAligned(), ... If DAC trigger is enabled, DAC conversion is performed only when a hardware or software trigger event is occurring. Select trigger source using function LL_DAC_SetTriggerSource().</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR TEN1 LL_DAC_EnableTrigger</li> <li>• CR TEN2 LL_DAC_EnableTrigger</li> </ul>                                                                                                                                                                                                                                                                                                                                              |

**LL\_DAC\_DisableTrigger**

|                      |                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DAC_DisableTrigger (DAC_TypeDef * DACx, uint32_t DAC_Channel)</code>                                                                                                                  |
| Function description | Disable DAC trigger of the selected channel.                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on</li> </ul> |

|                                                   |                                                                                                                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <p>all devices. Refer to device datasheet for channels availability.</p> <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR TEN1 LL_DAC_DisableTrigger</li> <li>• CR TEN2 LL_DAC_DisableTrigger</li> </ul>                                                    |

### LL\_DAC\_IsTriggerEnabled

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_DAC_IsTriggerEnabled(DAC_TypeDef * DACx, uint32_t DAC_Channel)</code></b>                                                                                                                                                                                                                                                                              |
| Function description                              | Get DAC trigger state of the selected channel.                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR TEN1 LL_DAC_IsTriggerEnabled</li> <li>• CR TEN2 LL_DAC_IsTriggerEnabled</li> </ul>                                                                                                                                                                                                                                                              |

### LL\_DAC\_TrigSWConversion

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_DAC_TrigSWConversion(DAC_TypeDef * DACx, uint32_t DAC_Channel)</code></b>                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description | Trig DAC conversion by software for the selected DAC channel.                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can a combination of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> </ul>                                                                                                 |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• Preliminarily, DAC trigger must be set to software trigger using function LL_DAC_SetTriggerSource() with parameter "LL_DAC_TRIGGER_SOFTWARE". and DAC trigger must be enabled using function LL_DAC_EnableTrigger().</li> <li>• For devices featuring DAC with 2 channels: this function can perform a SW start of both DAC channels simultaneously. Two channels can be selected as parameter. Example: (LL_DAC_CHANNEL_1   LL_DAC_CHANNEL_2)</li> </ul> |
| Reference Manual to  | <ul style="list-style-type: none"> <li>• SWTRIGR SWTRIG1 LL_DAC_TrigSWConversion</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                        |

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL API cross reference:                     | <ul style="list-style-type: none"> <li>• SWTRIGR SWTRIG2 LL_DAC_TrigSWConversion</li> </ul>                                                                                                                                                                                                                                                                                                                                                                          |
| <b>LL_DAC_ConvertData12RightAligned</b>     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function name                               | <b><code>__STATIC_INLINE void LL_DAC_ConvertData12RightAligned(DAC_TypeDef * DACx, uint32_t DAC_Channel, uint32_t Data)</code></b>                                                                                                                                                                                                                                                                                                                                   |
| Function description                        | Set the data to be loaded in the data holding register in format 12 bits left alignment (LSB aligned on bit 0), for the selected DAC channel.                                                                                                                                                                                                                                                                                                                        |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> <li>• <b>Data:</b> Value between Min_Data=0x000 and Max_Data=0xFFFF</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• DHR12R1 DACC1DHR LL_DAC_ConvertData12RightAligned</li> <li>• DHR12R2 DACC2DHR LL_DAC_ConvertData12RightAligned</li> </ul>                                                                                                                                                                                                                                                                                                   |

### LL\_DAC\_ConvertData12LeftAligned

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE void LL_DAC_ConvertData12LeftAligned(DAC_TypeDef * DACx, uint32_t DAC_Channel, uint32_t Data)</code></b>                                                                                                                                                                                                                                                                                                                                    |
| Function description                        | Set the data to be loaded in the data holding register in format 12 bits left alignment (MSB aligned on bit 15), for the selected DAC channel.                                                                                                                                                                                                                                                                                                                       |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> <li>• <b>Data:</b> Value between Min_Data=0x000 and Max_Data=0xFFFF</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• DHR12L1 DACC1DHR LL_DAC_ConvertData12LeftAligned</li> <li>• DHR12L2 DACC2DHR LL_DAC_ConvertData12LeftAligned</li> </ul>                                                                                                                                                                                                                                                                                                     |

**LL\_DAC\_ConvertData8RightAligned**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_DAC_ConvertData8RightAligned (DAC_TypeDef * DACx, uint32_t DAC_Channel, uint32_t Data)</code></b>                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Set the data to be loaded in the data holding register in format 8 bits left alignment (LSB aligned on bit 0), for the selected DAC channel.                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>- <code>LL_DAC_CHANNEL_1</code></li> <li>- <code>LL_DAC_CHANNEL_2</code> (1)</li> </ul> </li> <li>• <b>Data:</b> Value between Min_Data=0x00 and Max_Data=0xFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DHR8R1 DACC1DHR <code>LL_DAC_ConvertData8RightAligned</code></li> <li>• DHR8R2 DACC2DHR <code>LL_DAC_ConvertData8RightAligned</code></li> </ul>                                                                                                                                                                                                                                                                                                    |

**LL\_DAC\_ConvertDualData12RightAligned**

|                                                   |                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_DAC_ConvertDualData12RightAligned (DAC_TypeDef * DACx, uint32_t DataChannel1, uint32_t DataChannel2)</code></b>                                                                                                 |
| Function description                              | Set the data to be loaded in the data holding register in format 12 bits left alignment (LSB aligned on bit 0), for both DAC channels.                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DataChannel1:</b> Value between Min_Data=0x000 and Max_Data=0xFFFF</li> <li>• <b>DataChannel2:</b> Value between Min_Data=0x000 and Max_Data=0xFFFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DHR12RD DACC1DHR <code>LL_DAC_ConvertDualData12RightAligned</code></li> <li>• DHR12RD DACC2DHR <code>LL_DAC_ConvertDualData12RightAligned</code></li> </ul>                                            |

**LL\_DAC\_ConvertDualData12LeftAligned**

|                      |                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_DAC_ConvertDualData12LeftAligned (DAC_TypeDef * DACx, uint32_t DataChannel1, uint32_t DataChannel2)</code></b>                                                                                  |
| Function description | Set the data to be loaded in the data holding register in format 12 bits left alignment (MSB aligned on bit 15), for both DAC channels.                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DataChannel1:</b> Value between Min_Data=0x000 and Max_Data=0xFFFF</li> <li>• <b>DataChannel2:</b> Value between Min_Data=0x000 and</li> </ul> |

|                                                   |                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | Max_Data=0xFFFF                                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DHR12LD DACC1DHR<br/>LL_DAC_ConvertDualData12LeftAligned</li> <li>• DHR12LD DACC2DHR<br/>LL_DAC_ConvertDualData12LeftAligned</li> </ul> |

### LL\_DAC\_ConvertDualData8RightAligned

|                                                   |                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_DAC_ConvertDualData8RightAligned (DAC_TypeDef * DACx, uint32_t DataChannel1, uint32_t DataChannel2)</code></b>                                                                                           |
| Function description                              | Set the data to be loaded in the data holding register in format 8 bits left alignment (LSB aligned on bit 0), for both DAC channels.                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DataChannel1:</b> Value between Min_Data=0x00 and Max_Data=0xFF</li> <li>• <b>DataChannel2:</b> Value between Min_Data=0x00 and Max_Data=0xFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DHR8RD DACC1DHR<br/>LL_DAC_ConvertDualData8RightAligned</li> <li>• DHR8RD DACC2DHR<br/>LL_DAC_ConvertDualData8RightAligned</li> </ul>                                                            |

### LL\_DAC\_RetrieveOutputData

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_DAC_RetrieveOutputData (DAC_TypeDef * DACx, uint32_t DAC_Channel)</code></b>                                                                                                                                                                                                                                                                           |
| Function description                              | Retrieve output data currently generated for the selected DAC channel.                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability. <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x000 and Max_Data=0xFFFF</li> </ul>                                                                                                                                                                                                                                                                                |
| Notes                                             | <ul style="list-style-type: none"> <li>• Whatever alignment and resolution settings (using functions "LL_DAC_ConvertData{8; 12}{Right; Left} Aligned()": LL_DAC_ConvertData12RightAligned(), ...), output data format is 12 bits right aligned (LSB aligned on bit 0).</li> </ul>                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DOR1 DACC1DOR LL_DAC_RetrieveOutputData</li> <li>• DOR2 DACC2DOR LL_DAC_RetrieveOutputData</li> </ul>                                                                                                                                                                                                                                              |

**LL\_DAC\_IsActiveFlag\_DMAUDR1**

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DAC_IsActiveFlag_DMAUDR1(DAC_TypeDef * DACx)</code>    |
| Function description                              | Get DAC underrun flag for DAC channel 1.                                                 |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>DACx:</b> DAC instance</li></ul>              |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• SR DMAUDR1 LL_DAC_IsActiveFlag_DMAUDR1</li></ul> |

**LL\_DAC\_IsActiveFlag\_DMAUDR2**

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DAC_IsActiveFlag_DMAUDR2(DAC_TypeDef * DACx)</code>    |
| Function description                              | Get DAC underrun flag for DAC channel 2.                                                 |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>DACx:</b> DAC instance</li></ul>              |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• SR DMAUDR2 LL_DAC_IsActiveFlag_DMAUDR2</li></ul> |

**LL\_DAC\_ClearFlag\_DMAUDR1**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DAC_ClearFlag_DMAUDR1(DAC_TypeDef * DACx)</code>        |
| Function description                              | Clear DAC underrun flag for DAC channel 1.                                            |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>DACx:</b> DAC instance</li></ul>           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• SR DMAUDR1 LL_DAC_ClearFlag_DMAUDR1</li></ul> |

**LL\_DAC\_ClearFlag\_DMAUDR2**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DAC_ClearFlag_DMAUDR2(DAC_TypeDef * DACx)</code>        |
| Function description                              | Clear DAC underrun flag for DAC channel 2.                                            |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>DACx:</b> DAC instance</li></ul>           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• SR DMAUDR2 LL_DAC_ClearFlag_DMAUDR2</li></ul> |

**LL\_DAC\_EnableIT\_DMAUDR1**

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DAC_EnableIT_DMAUDR1(DAC_TypeDef * DACx)</code>            |
| Function description                              | Enable DMA underrun interrupt for DAC channel 1.                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR DMAUDRIE1 LL_DAC_EnableIT_DMAUDR1</li> </ul> |

**LL\_DAC\_EnableIT\_DMAUDR2**

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DAC_EnableIT_DMAUDR2(DAC_TypeDef * DACx)</code>            |
| Function description                              | Enable DMA underrun interrupt for DAC channel 2.                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR DMAUDRIE2 LL_DAC_EnableIT_DMAUDR2</li> </ul> |

**LL\_DAC\_DisableIT\_DMAUDR1**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DAC_DisableIT_DMAUDR1(DAC_TypeDef * DACx)</code>            |
| Function description                              | Disable DMA underrun interrupt for DAC channel 1.                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR DMAUDRIE1 LL_DAC_DisableIT_DMAUDR1</li> </ul> |

**LL\_DAC\_DisableIT\_DMAUDR2**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DAC_DisableIT_DMAUDR2(DAC_TypeDef * DACx)</code>            |
| Function description                              | Disable DMA underrun interrupt for DAC channel 2.                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR DMAUDRIE2 LL_DAC_DisableIT_DMAUDR2</li> </ul> |

**LL\_DAC\_IsEnabledIT\_DMAUDR1**

|                                                   |                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DAC_IsEnabledIT_DMAUDR1(DAC_TypeDef * DACx)</code>        |
| Function description                              | Get DMA underrun interrupt for DAC channel 1.                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR DMAUDRIE1 LL_DAC_IsEnabledIT_DMAUDR1</li> </ul> |

**LL\_DAC\_IsEnabledIT\_DMAUDR2**

|                                                   |                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DAC_IsEnabledIT_DMAUDR2(DAC_TypeDef * DACx)</code>        |
| Function description                              | Get DMA underrun interrupt for DAC channel 2.                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR DMAUDRIE2 LL_DAC_IsEnabledIT_DMAUDR2</li> </ul> |

**LL\_DAC\_DeInit**

|                      |                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_DAC_DeInit (DAC_TypeDef * DACx)</b>                                                                                                                                                                                   |
| Function description | De-initialize registers of the selected DAC instance to their default reset values.                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> </ul>                                                                                                                                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: DAC registers are de-initialized</li> <li>– ERROR: not applicable</li> </ul> </li> </ul> |

**LL\_DAC\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_DAC_Init (DAC_TypeDef * DACx, uint32_t DAC_Channel, LL_DAC_InitTypeDef * DAC_InitStruct)</b>                                                                                                                                                                                                                                                                                                                                                                    |
| Function description | Initialize some features of DAC instance.                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DACx:</b> DAC instance</li> <li>• <b>DAC_Channel:</b> This parameter can be one of the following values: (1) On this STM32 serie, parameter not available on all devices. Refer to device datasheet for channels availability.           <ul style="list-style-type: none"> <li>– LL_DAC_CHANNEL_1</li> <li>– LL_DAC_CHANNEL_2 (1)</li> </ul> </li> <li>• <b>DAC_InitStruct:</b> Pointer to a LL_DAC_InitTypeDef structure</li> </ul> |

|               |                                                                                                                                                                                                                                                   |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>SUCCESS: DAC registers are initialized</li> <li>ERROR: DAC registers are not initialized</li> </ul> </li> </ul> |
| Notes         | <ul style="list-style-type: none"> <li>The setting of these parameters by function LL_DAC_Init() is conditioned to DAC state: DAC instance must be disabled.</li> </ul>                                                                           |

### LL\_DAC\_StructInit

|                      |                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void LL_DAC_StructInit (LL_DAC_InitTypeDef * DAC_InitStruct)</code>                                                                                      |
| Function description | Set each LL_DAC_InitTypeDef field to default value.                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li><b>DAC_InitStruct:</b> pointer to a LL_DAC_InitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                  |

## 64.3 DAC Firmware driver defines

### 64.3.1 DAC

#### *DAC channels*

`LL_DAC_CHANNEL_1` DAC channel 1

`LL_DAC_CHANNEL_2` DAC channel 2

#### *DAC flags*

`LL_DAC_FLAG_DMAUDR1` DAC channel 1 flag DMA underrun

`LL_DAC_FLAG_DMAUDR2` DAC channel 2 flag DMA underrun

#### *Definitions of DAC hardware constraints delays*

`LL_DAC_DELAY_STARTUP_VOLTAGE_SETTLING_US` Delay for DAC channel voltage settling time from DAC channel startup (transition from disable to enable)

`LL_DAC_DELAY_VOLTAGE_SETTLING_US` Delay for DAC channel voltage settling time

#### *DAC interruptions*

`LL_DAC_IT_DMAUDRIE1` DAC channel 1 interruption DMA underrun

`LL_DAC_IT_DMAUDRIE2` DAC channel 2 interruption DMA underrun

#### *DAC channel output buffer*

`LL_DAC_OUTPUT_BUFFER_ENABLE` The selected DAC channel output is buffered: higher drive current capability, but also higher current consumption

`LL_DAC_OUTPUT_BUFFER_DISABLE` The selected DAC channel output is not buffered: lower drive current capability, but also lower current consumption

***DAC registers compliant with specific purpose***

|                                          |                                                         |
|------------------------------------------|---------------------------------------------------------|
| LL_DAC_DMA_REG_DATA_12BITS_RIGHT_ALIGNED | DAC channel data holding register 12 bits right aligned |
| LL_DAC_DMA_REG_DATA_12BITS_LEFT_ALIGNED  | DAC channel data holding register 12 bits left aligned  |
| LL_DAC_DMA_REG_DATA_8BITS_RIGHT_ALIGNED  | DAC channel data holding register 8 bits right aligned  |

***DAC channel output resolution***

|                       |                                |
|-----------------------|--------------------------------|
| LL_DAC_RESOLUTION_12B | DAC channel resolution 12 bits |
| LL_DAC_RESOLUTION_8B  | DAC channel resolution 8 bits  |

***DAC trigger source***

|                            |                                                                                                                                                                                                    |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_DAC_TRIG_SOFTWARE       | DAC channel conversion trigger internal (SW start)                                                                                                                                                 |
| LL_DAC_TRIG_EXT_TIM6_TRGO  | DAC channel conversion trigger from external IP: TIM6 TRGO.                                                                                                                                        |
| LL_DAC_TRIG_EXT_TIM3_TRGO  | DAC channel conversion trigger from external IP: TIM3 TRGO. Trigger remap: by default, default trigger. If needed to restore trigger, use LL_SYSCFG_DAC1_TRIG1_REMAP_TIM3_TRGO for TIM3 selection. |
| LL_DAC_TRIG_EXT_TIM7_TRGO  | DAC channel conversion trigger from external IP: TIM7 TRGO.                                                                                                                                        |
| LL_DAC_TRIG_EXT_TIM15_TRGO | DAC channel conversion trigger from external IP: TIM5 TRGO.                                                                                                                                        |
| LL_DAC_TRIG_EXT_TIM2_TRGO  | DAC channel conversion trigger from external IP: TIM2 TRGO.                                                                                                                                        |
| LL_DAC_TRIG_EXT_TIM4_TRGO  | DAC channel conversion trigger from external IP: TIM4 TRGO.                                                                                                                                        |
| LL_DAC_TRIG_EXT_TIM8_TRGO  | DAC channel conversion trigger from external IP: TIM8 TRGO. Trigger remap: use LL_SYSCFG_DAC1_TRIG1_REMAP_TIM8_TRGO for TIM8 selection.                                                            |
| LL_DAC_TRIG_EXT EXTI_LINE9 | DAC channel conversion trigger from external IP: external interrupt line 9.                                                                                                                        |

***DAC waveform automatic generation mode***

|                                      |                                                                                 |
|--------------------------------------|---------------------------------------------------------------------------------|
| LL_DAC_WAVE_AUTO_GENERATION_NONE     | DAC channel wave auto generation mode disabled.                                 |
| LL_DAC_WAVE_AUTO_GENERATION_NOISE    | DAC channel wave auto generation mode enabled, set generated noise waveform.    |
| LL_DAC_WAVE_AUTO_GENERATION_TRIANGLE | DAC channel wave auto generation mode enabled, set generated triangle waveform. |

**DAC wave generation - Noise LFSR unmask bits**

|                                   |                                                                             |
|-----------------------------------|-----------------------------------------------------------------------------|
| LL_DAC_NOISE_LFSR_UNMASK_BIT0     | Noise wave generation, unmask LFSR bit0, for the selected DAC channel       |
| LL_DAC_NOISE_LFSR_UNMASK_BITS1_0  | Noise wave generation, unmask LFSR bits[1:0], for the selected DAC channel  |
| LL_DAC_NOISE_LFSR_UNMASK_BITS2_0  | Noise wave generation, unmask LFSR bits[2:0], for the selected DAC channel  |
| LL_DAC_NOISE_LFSR_UNMASK_BITS3_0  | Noise wave generation, unmask LFSR bits[3:0], for the selected DAC channel  |
| LL_DAC_NOISE_LFSR_UNMASK_BITS4_0  | Noise wave generation, unmask LFSR bits[4:0], for the selected DAC channel  |
| LL_DAC_NOISE_LFSR_UNMASK_BITS5_0  | Noise wave generation, unmask LFSR bits[5:0], for the selected DAC channel  |
| LL_DAC_NOISE_LFSR_UNMASK_BITS6_0  | Noise wave generation, unmask LFSR bits[6:0], for the selected DAC channel  |
| LL_DAC_NOISE_LFSR_UNMASK_BITS7_0  | Noise wave generation, unmask LFSR bits[7:0], for the selected DAC channel  |
| LL_DAC_NOISE_LFSR_UNMASK_BITS8_0  | Noise wave generation, unmask LFSR bits[8:0], for the selected DAC channel  |
| LL_DAC_NOISE_LFSR_UNMASK_BITS9_0  | Noise wave generation, unmask LFSR bits[9:0], for the selected DAC channel  |
| LL_DAC_NOISE_LFSR_UNMASK_BITS10_0 | Noise wave generation, unmask LFSR bits[10:0], for the selected DAC channel |
| LL_DAC_NOISE_LFSR_UNMASK_BITS11_0 | Noise wave generation, unmask LFSR bits[11:0], for the selected DAC channel |

**DAC wave generation - Triangle amplitude**

|                               |                                                                                                 |
|-------------------------------|-------------------------------------------------------------------------------------------------|
| LL_DAC_TRIANGLE_AMPLITUDE_1   | Triangle wave generation, amplitude of 1 LSB of DAC output range, for the selected DAC channel  |
| LL_DAC_TRIANGLE_AMPLITUDE_3   | Triangle wave generation, amplitude of 3 LSB of DAC output range, for the selected DAC channel  |
| LL_DAC_TRIANGLE_AMPLITUDE_7   | Triangle wave generation, amplitude of 7 LSB of DAC output range, for the selected DAC channel  |
| LL_DAC_TRIANGLE_AMPLITUDE_15  | Triangle wave generation, amplitude of 15 LSB of DAC output range, for the selected DAC channel |
| LL_DAC_TRIANGLE_AMPLITUDE_31  | Triangle wave generation, amplitude of 31 LSB of DAC output range, for the selected DAC channel |
| LL_DAC_TRIANGLE_AMPLITUDE_63  | Triangle wave generation, amplitude of 63 LSB of DAC output range, for the selected DAC channel |
| LL_DAC_TRIANGLE_AMPLITUDE_127 | Triangle wave generation, amplitude of 127                                                      |

|                                |                                                                                                      |
|--------------------------------|------------------------------------------------------------------------------------------------------|
|                                | LSB of DAC output range, for the selected DAC channel                                                |
| LL_DAC_TRIANGLE_AMPLITUDE_255  | Triangle wave generation, amplitude of 255<br>LSB of DAC output range, for the selected DAC channel  |
| LL_DAC_TRIANGLE_AMPLITUDE_511  | Triangle wave generation, amplitude of 512<br>LSB of DAC output range, for the selected DAC channel  |
| LL_DAC_TRIANGLE_AMPLITUDE_1023 | Triangle wave generation, amplitude of 1023<br>LSB of DAC output range, for the selected DAC channel |
| LL_DAC_TRIANGLE_AMPLITUDE_2047 | Triangle wave generation, amplitude of 2047<br>LSB of DAC output range, for the selected DAC channel |
| LL_DAC_TRIANGLE_AMPLITUDE_4095 | Triangle wave generation, amplitude of 4095<br>LSB of DAC output range, for the selected DAC channel |

**DAC helper macro**\_LL\_DAC\_CHANNEL\_TO\_DECIMAL\_NB**Description:**

- Helper macro to get DAC channel number in decimal format from literals LL\_DAC\_CHANNEL\_x.

**Parameters:**

- \_\_CHANNEL\_\_: This parameter can be one of the following values:
  - LL\_DAC\_CHANNEL\_1
  - LL\_DAC\_CHANNEL\_2 (1)

**Return value:**

- 1...2: (value "2" depending on DAC channel 2 availability)

**Notes:**

- The input can be a value from functions where a channel number is returned.

\_LL\_DAC\_DECIMAL\_NB\_TO\_CHANNEL**Description:**

- Helper macro to get DAC channel in literal format LL\_DAC\_CHANNEL\_x from number in decimal format.

**Parameters:**

- \_\_DECIMAL\_NB\_\_: 1...2 (value "2" depending on DAC channel 2 availability)

**Return value:**

- Returned: value can be one of the following values:
  - LL\_DAC\_CHANNEL\_1

- LL\_DAC\_CHANNEL\_2 (1)

**Notes:**

- If the input parameter does not correspond to a DAC channel, this macro returns value '0'.

### \_\_LL\_DAC\_DIGITAL\_SCALE

**Description:**

- Helper macro to define the DAC conversion data full-scale digital value corresponding to the selected DAC resolution.

**Parameters:**

- \_\_DAC\_RESOLUTION\_\_: This parameter can be one of the following values:
  - LL\_DAC\_RESOLUTION\_12B
  - LL\_DAC\_RESOLUTION\_8B

**Return value:**

- ADC: conversion data equivalent voltage value (unit: mVolt)

**Notes:**

- DAC conversion data full-scale corresponds to voltage range determined by analog voltage references Vref+ and Vref- (refer to reference manual).

### \_\_LL\_DAC\_CALC\_VOLTAGE\_TO\_DATA

**Description:**

- Helper macro to calculate the DAC conversion data (unit: digital value) corresponding to a voltage (unit: mVolt).

**Parameters:**

- \_\_VREFANALOG\_VOLTAGE\_\_: Analog reference voltage (unit: mV)
- \_\_DAC\_VOLTAGE\_\_: Voltage to be generated by DAC channel (unit: mVolt).
- \_\_DAC\_RESOLUTION\_\_: This parameter can be one of the following values:
  - LL\_DAC\_RESOLUTION\_12B
  - LL\_DAC\_RESOLUTION\_8B

**Return value:**

- DAC: conversion data (unit: digital value)

**Notes:**

- This helper macro is intended to provide input data in voltage rather than digital value, to be used with LL DAC functions such as LL\_DAC\_ConvertData12RightAligned(). Analog reference voltage (Vref+) must be

either known from user board environment or can be calculated using ADC measurement and ADC helper macro  
`__LL_ADC_CALC_VREFANALOG_VOLTAGE()`.

#### ***Common write and read registers macros***

`LL_DAC_WriteReg`

##### **Description:**

- Write a value in DAC register.

##### **Parameters:**

- `__INSTANCE__`: DAC Instance
- `__REG__`: Register to be written
- `__VALUE__`: Value to be written in the register

##### **Return value:**

- None

`LL_DAC_ReadReg`

##### **Description:**

- Read a value in DAC register.

##### **Parameters:**

- `__INSTANCE__`: DAC Instance
- `__REG__`: Register to be read

##### **Return value:**

- Register: value

## 65 LL DMA Generic Driver

### 65.1 DMA Firmware driver registers structures

#### 65.1.1 LL\_DMA\_InitTypeDef

##### Data Fields

- *uint32\_t PeriphOrM2MSrcAddress*
- *uint32\_t MemoryOrM2MDstAddress*
- *uint32\_t Direction*
- *uint32\_t Mode*
- *uint32\_t PeriphOrM2MSrcIncMode*
- *uint32\_t MemoryOrM2MDstIncMode*
- *uint32\_t PeriphOrM2MSrcDataSize*
- *uint32\_t MemoryOrM2MDstDataSize*
- *uint32\_t NbData*
- *uint32\_t Priority*

##### Field Documentation

- ***uint32\_t LL\_DMA\_InitTypeDef::PeriphOrM2MSrcAddress***  
Specifies the peripheral base address for DMA transfer or as Source base address in case of memory to memory transfer direction. This parameter must be a value between Min\_Data = 0 and Max\_Data = 0xFFFFFFFF.
- ***uint32\_t LL\_DMA\_InitTypeDef::MemoryOrM2MDstAddress***  
Specifies the memory base address for DMA transfer or as Destination base address in case of memory to memory transfer direction. This parameter must be a value between Min\_Data = 0 and Max\_Data = 0xFFFFFFFF.
- ***uint32\_t LL\_DMA\_InitTypeDef::Direction***  
Specifies if the data will be transferred from memory to peripheral, from memory to memory or from peripheral to memory. This parameter can be a value of **DMA\_LL\_EC\_DIRECTION** This feature can be modified afterwards using unitary function **LL\_DMA\_SetDataTransferDirection()**.
- ***uint32\_t LL\_DMA\_InitTypeDef::Mode***  
Specifies the normal or circular operation mode. This parameter can be a value of **DMA\_LL\_EC\_MODE**  
**Note:** The circular buffer mode cannot be used if the memory to memory data transfer direction is configured on the selected Channel This feature can be modified afterwards using unitary function **LL\_DMA\_SetMode()**.
- ***uint32\_t LL\_DMA\_InitTypeDef::PeriphOrM2MSrcIncMode***  
Specifies whether the Peripheral address or Source address in case of memory to memory transfer direction is incremented or not. This parameter can be a value of **DMA\_LL\_EC\_PERIPH** This feature can be modified afterwards using unitary function **LL\_DMA\_SetPeriphIncMode()**.
- ***uint32\_t LL\_DMA\_InitTypeDef::MemoryOrM2MDstIncMode***  
Specifies whether the Memory address or Destination address in case of memory to memory transfer direction is incremented or not. This parameter can be a value of **DMA\_LL\_EC\_MEMORY** This feature can be modified afterwards using unitary function **LL\_DMA\_SetMemoryIncMode()**.
- ***uint32\_t LL\_DMA\_InitTypeDef::PeriphOrM2MSrcDataSize***  
Specifies the Peripheral data size alignment or Source data size alignment (byte, half word, word) in case of memory to memory transfer direction. This parameter can be a

- value of **DMA\_LL\_EC\_PDATAALIGN**This feature can be modified afterwards using unitary function **LL\_DMA\_SetPeriphSize()**.
- **uint32\_t LL\_DMA\_InitTypeDef::MemoryOrM2MDstDataSize**  
Specifies the Memory data size alignment or Destination data size alignment (byte, half word, word) in case of memory to memory transfer direction. This parameter can be a value of **DMA\_LL\_EC\_MDATAALIGN**This feature can be modified afterwards using unitary function **LL\_DMA\_SetMemorySize()**.
  - **uint32\_t LL\_DMA\_InitTypeDef::NbData**  
Specifies the number of data to transfer, in data unit. The data unit is equal to the source buffer configuration set in PeripheralSize or MemorySize parameters depending in the transfer direction. This parameter must be a value between Min\_Data = 0 and Max\_Data = 0x0000FFFFThis feature can be modified afterwards using unitary function **LL\_DMA\_SetDataLength()**.
  - **uint32\_t LL\_DMA\_InitTypeDef::Priority**  
Specifies the channel priority level. This parameter can be a value of **DMA\_LL\_EC\_PRIORITY**This feature can be modified afterwards using unitary function **LL\_DMA\_SetChannelPriorityLevel()**.

## 65.2 DMA Firmware driver API description

### 65.2.1 Detailed description of functions

#### LL\_DMA\_EnableChannel

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DMA_EnableChannel<br/>(DMA_TypeDef * DMax, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                          |
| Function description                              | Enable DMA channel.                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMax:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR EN LL_DMA_EnableChannel</li> </ul>                                                                                                                                                                                                                                                                                                                            |

#### LL\_DMA\_DisableChannel

|                      |                                                                                                                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_DisableChannel<br/>(DMA_TypeDef * DMax, uint32_t Channel)</code>                                                                                                                                                                                         |
| Function description | Disable DMA channel.                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMax:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> </ul> </li> </ul> |

---

|                                                   |                                                                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR EN LL_DMA_DisableChannel</li> </ul>                                                                         |

### LL\_DMA\_IsEnabledChannel

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DMA_IsEnabledChannel(<br/>DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                   |
| Function description                              | Check if DMA channel is enabled or disabled.                                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR EN LL_DMA_IsEnabledChannel</li> </ul>                                                                                                                                                                                                                                                                                                                         |

### LL\_DMA\_ConfigTransfer

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_ConfigTransfer(<br/>DMA_TypeDef * DMAx, uint32_t Channel, uint32_t Configuration)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Configure all parameters link to DMA transfer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>Configuration:</b> This parameter must be a combination of all the following values: <ul style="list-style-type: none"> <li>- LL_DMA_DIRECTION_PERIPH_TO_MEMORY or<br/>LL_DMA_DIRECTION_MEMORY_TO_PERIPH or<br/>LL_DMA_DIRECTION_MEMORY_TO_MEMORY</li> <li>- LL_DMA_MODE_NORMAL or</li> </ul> </li> </ul> |

- LL\_DMA\_MODE\_CIRCULAR
- LL\_DMA\_PERIPH\_INCREMENT or  
LL\_DMA\_PERIPH\_NOINCREMENT
- LL\_DMA\_MEMORY\_INCREMENT or  
LL\_DMA\_MEMORY\_NOINCREMENT
- LL\_DMA\_PDATAALIGN\_BYTE or  
LL\_DMA\_PDATAALIGN\_HALFWORD or  
LL\_DMA\_PDATAALIGN\_WORD
- LL\_DMA\_MDATAALIGN\_BYTE or  
LL\_DMA\_MDATAALIGN\_HALFWORD or  
LL\_DMA\_MDATAALIGN\_WORD
- LL\_DMA\_PRIORITY\_LOW or  
LL\_DMA\_PRIORITY\_MEDIUM or  
LL\_DMA\_PRIORITY\_HIGH or  
LL\_DMA\_PRIORITY\_VERYHIGH

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CCR DIR LL\_DMA\_ConfigTransfer
- CCR MEM2MEM LL\_DMA\_ConfigTransfer
- CCR CIRC LL\_DMA\_ConfigTransfer
- CCR PINC LL\_DMA\_ConfigTransfer
- CCR MINC LL\_DMA\_ConfigTransfer
- CCR PSIZE LL\_DMA\_ConfigTransfer
- CCR MSIZE LL\_DMA\_ConfigTransfer
- CCR PL LL\_DMA\_ConfigTransfer

### **LL\_DMA\_SetDataTransferDirection**

Function name

```
STATIC_INLINE void LL_DMA_SetDataTransferDirection
(DMA_TypeDef * DMAx, uint32_t Channel, uint32_t Direction)
```

Function description

Set Data transfer direction (read from peripheral or from memory).

Parameters

- **DMAx:** DMAx Instance
- **Channel:** This parameter can be one of the following values:
  - LL\_DMA\_CHANNEL\_1
  - LL\_DMA\_CHANNEL\_2
  - LL\_DMA\_CHANNEL\_3
  - LL\_DMA\_CHANNEL\_4
  - LL\_DMA\_CHANNEL\_5
  - LL\_DMA\_CHANNEL\_6
  - LL\_DMA\_CHANNEL\_7
- **Direction:** This parameter can be one of the following values:
  - LL\_DMA\_DIRECTION\_PERIPH\_TO\_MEMORY
  - LL\_DMA\_DIRECTION\_MEMORY\_TO\_PERIPH
  - LL\_DMA\_DIRECTION\_MEMORY\_TO\_MEMORY

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CCR DIR LL\_DMA\_SetDataTransferDirection
- CCR MEM2MEM LL\_DMA\_SetDataTransferDirection

**LL\_DMA\_GetDataTransferDirection**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_DMA_GetDataTransferDirection (DMA_TypeDef * DMAx,<br/>uint32_t Channel)</code>                                                                                                                                                                                                                                                                                       |
| Function description                              | Get Data transfer direction (read from peripheral or from memory).                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_DIRECTION_PERIPH_TO_MEMORY</li> <li>- LL_DMA_DIRECTION_MEMORY_TO_PERIPH</li> <li>- LL_DMA_DIRECTION_MEMORY_TO_MEMORY</li> </ul> </li> </ul>                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR DIR LL_DMA_GetDataTransferDirection</li> <li>• CCR MEM2MEM LL_DMA_GetDataTransferDirection</li> </ul>                                                                                                                                                                                                                                                         |

**LL\_DMA\_SetMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DMA_SetMode (DMA_TypeDef *<br/>DMAx, uint32_t Channel, uint32_t Mode)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description                              | Set DMA mode circular or normal.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>Mode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_MODE_NORMAL</li> <li>- LL_DMA_MODE_CIRCULAR</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• The circular buffer mode cannot be used if the memory-to-memory data transfer is configured on the selected Channel.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR CIRC LL_DMA_SetMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

**LL\_DMA\_GetMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DMA_GetMode (DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                |
| Function description                              | Get DMA mode circular or normal.                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_MODE_NORMAL</li> <li>– LL_DMA_MODE_CIRCULAR</li> </ul> </li> </ul>                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR CIRC LL_DMA_GetMode</li> </ul>                                                                                                                                                                                                                                                                                                                                |

**LL\_DMA\_SetPeriphIncMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DMA_SetPeriphIncMode (DMA_TypeDef * DMAx, uint32_t Channel, uint32_t PeriphOrM2MSrcIncMode)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description                              | Set Peripheral increment mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>PeriphOrM2MSrcIncMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_PERIPH_INCREMENT</li> <li>– LL_DMA_PERIPH_NOINCREMENT</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR PINC LL_DMA_SetPeriphIncMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

**LL\_DMA\_GetPeriphIncMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DMA_GetPeriphIncMode(DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                        |
| Function description                              | Get Peripheral increment mode.                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_PERIPH_INCREMENT</li> <li>– LL_DMA_PERIPH_NOINCREMENT</li> </ul> </li> </ul>                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR PINC LL_DMA_GetPeriphIncMode</li> </ul>                                                                                                                                                                                                                                                                                                                       |

**LL\_DMA\_SetMemoryIncMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DMA_SetMemoryIncMode(DMA_TypeDef * DMAx, uint32_t Channel, uint32_t MemoryOrM2MDstIncMode)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Set Memory increment mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>MemoryOrM2MDstIncMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_MEMORY_INCREMENT</li> <li>– LL_DMA_MEMORY_NOINCREMENT</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR MINC LL_DMA_SetMemoryIncMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

**LL\_DMA\_GetMemoryIncMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DMA_GetMemoryIncMode(DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                        |
| Function description                              | Get Memory increment mode.                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_MEMORY_INCREMENT</li> <li>– LL_DMA_MEMORY_NOINCREMENT</li> </ul> </li> </ul>                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR MINC LL_DMA_GetMemoryIncMode</li> </ul>                                                                                                                                                                                                                                                                                                                       |

**LL\_DMA\_SetPeriphSize**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DMA_SetPeriphSize(DMA_TypeDef * DMAx, uint32_t Channel, uint32_t PeriphOrM2MSrcDataSize)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Set Peripheral size.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>PeriphOrM2MSrcDataSize:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_PDATAALIGN_BYTE</li> <li>– LL_DMA_PDATAALIGN_HALFWORD</li> <li>– LL_DMA_PDATAALIGN_WORD</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR PSIZE LL_DMA_SetPeriphSize</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

**LL\_DMA\_GetPeriphSize**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_DMA_GetPeriphSize<br/>(DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                       |
| Function description                              | Get Peripheral size.                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_PDATAALIGN_BYTE</li> <li>– LL_DMA_PDATAALIGN_HALFWORD</li> <li>– LL_DMA_PDATAALIGN_WORD</li> </ul> </li> </ul>                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR PSIZE LL_DMA_GetPeriphSize</li> </ul>                                                                                                                                                                                                                                                                                                                         |

**LL\_DMA\_SetMemorySize**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_DMA_SetMemorySize<br/>(DMA_TypeDef * DMAx, uint32_t Channel, uint32_t<br/>MemoryOrM2MDstDataSize)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description                              | Set Memory size.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>MemoryOrM2MDstDataSize:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_MDATAALIGN_BYTE</li> <li>– LL_DMA_MDATAALIGN_HALFWORD</li> <li>– LL_DMA_MDATAALIGN_WORD</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR MSIZE LL_DMA_SetMemorySize</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

**LL\_DMA\_GetMemorySize**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_DMA_GetMemorySize(DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                            |
| Function description                              | Get Memory size.                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_MDATAALIGN_BYTE</li> <li>- LL_DMA_MDATAALIGN_HALFWORD</li> <li>- LL_DMA_MDATAALIGN_WORD</li> </ul> </li> </ul>                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR MSIZE LL_DMA_GetMemorySize</li> </ul>                                                                                                                                                                                                                                                                                                                         |

**LL\_DMA\_SetChannelPriorityLevel**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_DMA_SetChannelPriorityLevel(DMA_TypeDef * DMAx, uint32_t Channel, uint32_t Priority)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description                              | Set Channel priority level.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>Priority:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_PRIORITY_LOW</li> <li>- LL_DMA_PRIORITY_MEDIUM</li> <li>- LL_DMA_PRIORITY_HIGH</li> <li>- LL_DMA_PRIORITY_VERYHIGH</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR PL LL_DMA_SetChannelPriorityLevel</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

**LL\_DMA\_GetChannelPriorityLevel**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DMA_GetChannelPriorityLevel(DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                 |
| Function description                              | Get Channel priority level.                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_PRIORITY_LOW</li> <li>- LL_DMA_PRIORITY_MEDIUM</li> <li>- LL_DMA_PRIORITY_HIGH</li> <li>- LL_DMA_PRIORITY_VERYHIGH</li> </ul> </li> </ul>                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR PL LL_DMA_GetChannelPriorityLevel</li> </ul>                                                                                                                                                                                                                                                                                                                  |

**LL\_DMA\_SetDataLength**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DMA_SetDataLength(DMA_TypeDef * DMAx, uint32_t Channel, uint32_t NbData)</code>                                                                                                                                                                                                                                                                                                                                                                       |
| Function description                              | Set Number of data to transfer.                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>NbData:</b> Between Min_Data = 0 and Max_Data = 0x0000FFFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• This action has no effect if channel is enabled.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CNDTR NDT LL_DMA_SetDataLength</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                  |

**LL\_DMA\_GetDataLength**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DMA_GetDataLength(DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                           |
| Function description                              | Get Number of data to transfer.                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Between:</b> Min_Data = 0 and Max_Data = 0xFFFFFFFF</li> </ul>                                                                                                                                                                                                                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>• Once the channel is enabled, the return value indicate the remaining bytes to be transmitted.</li> </ul>                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CNDTR NDT LL_DMA_GetDataLength</li> </ul>                                                                                                                                                                                                                                                                                                                         |

**LL\_DMA\_ConfigAddresses**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_ConfigAddresses(DMA_TypeDef * DMAx, uint32_t Channel, uint32_t SrcAddress, uint32_t DstAddress, uint32_t Direction)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Configure the Source and Destination addresses.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>SrcAddress:</b> Between Min_Data = 0 and Max_Data = 0xFFFFFFFF</li> <li>• <b>DstAddress:</b> Between Min_Data = 0 and Max_Data = 0xFFFFFFFF</li> <li>• <b>Direction:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_DIRECTION_PERIPH_TO_MEMORY</li> <li>- LL_DMA_DIRECTION_MEMORY_TO_PERIPH</li> <li>- LL_DMA_DIRECTION_MEMORY_TO_MEMORY</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• This API must not be called when the DMA channel is enabled.</li> <li>• Each IP using DMA provides an API to get directly the</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

register address (LL\_PPP\_DMA\_GetRegAddr).

Reference Manual to  
LL API cross  
reference:

- CPAR PA LL\_DMA\_ConfigAddresses
- CMAR MA LL\_DMA\_ConfigAddresses

### **LL\_DMA\_SetMemoryAddress**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DMA_SetMemoryAddress<br/>(DMA_TypeDef * DMAx, uint32_t Channel, uint32_t<br/>MemoryAddress)</code>                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Set the Memory address.                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>MemoryAddress:</b> Between Min_Data = 0 and Max_Data = 0xFFFFFFFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• Interface used for direction LL_DMA_DIRECTION_PERIPH_TO_MEMORY or LL_DMA_DIRECTION_MEMORY_TO_PERIPH only.</li> <li>• This API must not be called when the DMA channel is enabled.</li> </ul>                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CMAR MA LL_DMA_SetMemoryAddress</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                        |

### **LL\_DMA\_SetPeriphAddress**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_SetPeriphAddress<br/>(DMA_TypeDef * DMAx, uint32_t Channel, uint32_t<br/>PeriphAddress)</code>                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Set the Peripheral address.                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> <li>• <b>PeriphAddress:</b> Between Min_Data = 0 and Max_Data = 0xFFFFFFFF</li> </ul> |

|                                                   |                                                                                                                                                                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>Interface used for direction LL_DMA_DIRECTION_PERIPH_TO_MEMORY or LL_DMA_DIRECTION_MEMORY_TO_PERIPH only.</li> <li>This API must not be called when the DMA channel is enabled.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CPAR PA LL_DMA_SetPeriphAddress</li> </ul>                                                                                                                                                 |

### LL\_DMA\_GetMemoryAddress

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_DMA_GetMemoryAddress<br/>(DMA_TypeDef * DMAX, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                 |
| Function description                              | Get Memory address.                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>DMAX:</b> DMAx Instance</li> <li><b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_DMA_CHANNEL_1</li> <li>LL_DMA_CHANNEL_2</li> <li>LL_DMA_CHANNEL_3</li> <li>LL_DMA_CHANNEL_4</li> <li>LL_DMA_CHANNEL_5</li> <li>LL_DMA_CHANNEL_6</li> <li>LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Between:</b> Min_Data = 0 and Max_Data = 0xFFFFFFFF</li> </ul>                                                                                                                                                                                                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>Interface used for direction LL_DMA_DIRECTION_PERIPH_TO_MEMORY or LL_DMA_DIRECTION_MEMORY_TO_PERIPH only.</li> </ul>                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CMAR MA LL_DMA_GetMemoryAddress</li> </ul>                                                                                                                                                                                                                                                                                                        |

### LL\_DMA\_GetPeriphAddress

|                      |                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DMA_GetPeriphAddress<br/>(DMA_TypeDef * DMAX, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                 |
| Function description | Get Peripheral address.                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>DMAX:</b> DMAx Instance</li> <li><b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_DMA_CHANNEL_1</li> <li>LL_DMA_CHANNEL_2</li> <li>LL_DMA_CHANNEL_3</li> <li>LL_DMA_CHANNEL_4</li> <li>LL_DMA_CHANNEL_5</li> <li>LL_DMA_CHANNEL_6</li> <li>LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>Between:</b> Min_Data = 0 and Max_Data = 0xFFFFFFFF</li> </ul>                                                                                                                                                                                                                                                                                 |

---

|                                                   |                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                             | <ul style="list-style-type: none"> <li>Interface used for direction LL_DMA_DIRECTION_PERIPH_TO_MEMORY or LL_DMA_DIRECTION_MEMORY_TO_PERIPH only.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CPAR PA LL_DMA_GetPeriphAddress</li> </ul>                                                                           |

### LL\_DMA\_SetM2MSrcAddress

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DMA_SetM2MSrcAddress<br/>(DMA_TypeDef * DMAX, uint32_t Channel, uint32_t<br/>MemoryAddress)</code>                                                                                                                                                                                                                                                                                                                                                     |
| Function description                              | Set the Memory to Memory Source address.                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>DMAX:</b> DMAx Instance</li> <li><b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> <li><b>MemoryAddress:</b> Between Min_Data = 0 and Max_Data = 0xFFFFFFFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>Interface used for direction LL_DMA_DIRECTION_MEMORY_TO_MEMORY only.</li> <li>This API must not be called when the DMA channel is enabled.</li> </ul>                                                                                                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CPAR PA LL_DMA_SetM2MSrcAddress</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                    |

### LL\_DMA\_SetM2MDstAddress

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_SetM2MDstAddress<br/>(DMA_TypeDef * DMAX, uint32_t Channel, uint32_t<br/>MemoryAddress)</code>                                                                                                                                                                                                                                                                                                                                          |
| Function description | Set the Memory to Memory Destination address.                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li><b>DMAX:</b> DMAx Instance</li> <li><b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> <li><b>MemoryAddress:</b> Between Min_Data = 0 and Max_Data =</li> </ul> |

0xFFFFFFFF

|                                             |                                                                                                                                                                                              |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                |
| Notes                                       | <ul style="list-style-type: none"> <li>Interface used for direction LL_DMA_DIRECTION_MEMORY_TO_MEMORY only.</li> <li>This API must not be called when the DMA channel is enabled.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CMAR MA LL_DMA_SetM2MDstAddress</li> </ul>                                                                                                            |

### LL\_DMA\_GetM2MSrcAddress

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_DMA_GetM2MSrcAddress(DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                    |
| Function description                        | Get the Memory to Memory Source address.                                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>DMAx:</b> DMAx Instance</li> <li><b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>Between:</b> Min_Data = 0 and Max_Data = 0xFFFFFFFF</li> </ul>                                                                                                                                                                                                                                                                                               |
| Notes                                       | <ul style="list-style-type: none"> <li>Interface used for direction LL_DMA_DIRECTION_MEMORY_TO_MEMORY only.</li> </ul>                                                                                                                                                                                                                                                                                 |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CPAR PA LL_DMA_GetM2MSrcAddress</li> </ul>                                                                                                                                                                                                                                                                                                                      |

### LL\_DMA\_GetM2MDstAddress

|                      |                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DMA_GetM2MDstAddress(DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                    |
| Function description | Get the Memory to Memory Destination address.                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>DMAx:</b> DMAx Instance</li> <li><b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>Between:</b> Min_Data = 0 and Max_Data = 0xFFFFFFFF</li> </ul>                                                                                                                                                                                                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>Interface used for direction</li> </ul>                                                                                                                                                                                                                                                                                                                         |

Reference Manual to  
LL API cross  
reference:

LL\_DMA\_DIRECTION\_MEMORY\_TO\_MEMORY only.

- CMAR MA LL\_DMA\_GetM2MDstAddress

### **LL\_DMA\_IsActiveFlag\_GI1**

Function name      **\_STATIC\_INLINE uint32\_t LL\_DMA\_IsActiveFlag\_GI1  
(DMA\_TypeDef \* DMAx)**

Function description      Get Channel 1 global interrupt flag.

Parameters      • **DMAx**: DMAx Instance

Return values      • **State**: of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
ISR GIF1 LL\_DMA\_IsActiveFlag\_GI1

### **LL\_DMA\_IsActiveFlag\_GI2**

Function name      **\_STATIC\_INLINE uint32\_t LL\_DMA\_IsActiveFlag\_GI2  
(DMA\_TypeDef \* DMAx)**

Function description      Get Channel 2 global interrupt flag.

Parameters      • **DMAx**: DMAx Instance

Return values      • **State**: of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
ISR GIF2 LL\_DMA\_IsActiveFlag\_GI2

### **LL\_DMA\_IsActiveFlag\_GI3**

Function name      **\_STATIC\_INLINE uint32\_t LL\_DMA\_IsActiveFlag\_GI3  
(DMA\_TypeDef \* DMAx)**

Function description      Get Channel 3 global interrupt flag.

Parameters      • **DMAx**: DMAx Instance

Return values      • **State**: of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
ISR GIF3 LL\_DMA\_IsActiveFlag\_GI3

### **LL\_DMA\_IsActiveFlag\_GI4**

Function name      **\_STATIC\_INLINE uint32\_t LL\_DMA\_IsActiveFlag\_GI4  
(DMA\_TypeDef \* DMAx)**

Function description      Get Channel 4 global interrupt flag.

Parameters      • **DMAx**: DMAx Instance

Return values      • **State**: of bit (1 or 0).

- Reference Manual to  
LL API cross  
reference:
- ISR GIF4 LL\_DMA\_IsActiveFlag\_GI4

### LL\_DMA\_IsActiveFlag\_GI5

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_GI5(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 5 global interrupt flag.                                              |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>      |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>  |

Reference Manual to  
LL API cross  
reference:

### LL\_DMA\_IsActiveFlag\_GI6

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_GI6(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 6 global interrupt flag.                                              |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>      |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>  |

Reference Manual to  
LL API cross  
reference:

### LL\_DMA\_IsActiveFlag\_GI7

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_GI7(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 7 global interrupt flag.                                              |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>      |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>  |

Reference Manual to  
LL API cross  
reference:

### LL\_DMA\_IsActiveFlag\_TC1

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TC1(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 1 transfer complete flag.                                             |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>      |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>  |

- Reference Manual to • ISR TCIF1 LL\_DMA\_IsActiveFlag\_TC1  
 LL API cross reference:

### **LL\_DMA\_IsActiveFlag\_TC2**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TC2(DMA_TypeDef * DMAx)</code></b> |
| Function description | Get Channel 2 transfer complete flag.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> </ul>          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>      |

- Reference Manual to • ISR TCIF2 LL\_DMA\_IsActiveFlag\_TC2  
 LL API cross reference:

### **LL\_DMA\_IsActiveFlag\_TC3**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TC3(DMA_TypeDef * DMAx)</code></b> |
| Function description | Get Channel 3 transfer complete flag.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> </ul>          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>      |

- Reference Manual to • ISR TCIF3 LL\_DMA\_IsActiveFlag\_TC3  
 LL API cross reference:

### **LL\_DMA\_IsActiveFlag\_TC4**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TC4(DMA_TypeDef * DMAx)</code></b> |
| Function description | Get Channel 4 transfer complete flag.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> </ul>          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>      |

- Reference Manual to • ISR TCIF4 LL\_DMA\_IsActiveFlag\_TC4  
 LL API cross reference:

### **LL\_DMA\_IsActiveFlag\_TC5**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TC5(DMA_TypeDef * DMAx)</code></b> |
| Function description | Get Channel 5 transfer complete flag.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> </ul>          |

- Return values

- Reference Manual to  
LL API cross  
reference:
- ISR TCIF5 LL\_DMA\_IsActiveFlag\_TC5

### LL\_DMA\_IsActiveFlag\_TC6

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TC6<br/>(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 6 transfer complete flag.                                                  |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>           |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>       |

Reference Manual to  
LL API cross  
reference:

### LL\_DMA\_IsActiveFlag\_TC7

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TC7<br/>(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 7 transfer complete flag.                                                  |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>           |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>       |

Reference Manual to  
LL API cross  
reference:

### LL\_DMA\_IsActiveFlag\_HT1

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_HT1<br/>(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 1 half transfer flag.                                                      |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>           |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>       |

Reference Manual to  
LL API cross  
reference:

### LL\_DMA\_IsActiveFlag\_HT2

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_HT2<br/>(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 2 half transfer flag.                                                      |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>           |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>       |

Reference Manual to  
LL API cross  
reference:

- ISR HTIF2 LL\_DMA\_IsActiveFlag\_HT2

### **LL\_DMA\_IsActiveFlag\_HT3**

Function name **\_STATIC\_INLINE uint32\_t LL\_DMA\_IsActiveFlag\_HT3**

**(DMA\_TypeDef \* DMAx)**

Function description Get Channel 3 half transfer flag.

Parameters • **DMAx:** DMAx Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR HTIF3 LL\_DMA\_IsActiveFlag\_HT3

### **LL\_DMA\_IsActiveFlag\_HT4**

Function name **\_STATIC\_INLINE uint32\_t LL\_DMA\_IsActiveFlag\_HT4**

**(DMA\_TypeDef \* DMAx)**

Function description Get Channel 4 half transfer flag.

Parameters • **DMAx:** DMAx Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR HTIF4 LL\_DMA\_IsActiveFlag\_HT4

### **LL\_DMA\_IsActiveFlag\_HT5**

Function name **\_STATIC\_INLINE uint32\_t LL\_DMA\_IsActiveFlag\_HT5**

**(DMA\_TypeDef \* DMAx)**

Function description Get Channel 5 half transfer flag.

Parameters • **DMAx:** DMAx Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR HTIF5 LL\_DMA\_IsActiveFlag\_HT5

### **LL\_DMA\_IsActiveFlag\_HT6**

Function name **\_STATIC\_INLINE uint32\_t LL\_DMA\_IsActiveFlag\_HT6**

**(DMA\_TypeDef \* DMAx)**

Function description Get Channel 6 half transfer flag.

Parameters • **DMAx:** DMAx Instance

Return values • **State:** of bit (1 or 0).

- Reference Manual to  
LL API cross  
reference:
- ISR HTIF6 LL\_DMA\_IsActiveFlag\_HT6

### LL\_DMA\_IsActiveFlag\_HT7

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_HT7<br/>(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 7 half transfer flag.                                                     |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>          |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>      |

Reference Manual to  
LL API cross  
reference:

### LL\_DMA\_IsActiveFlag\_TE1

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TE1<br/>(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 1 transfer error flag.                                                    |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>          |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>      |

Reference Manual to  
LL API cross  
reference:

### LL\_DMA\_IsActiveFlag\_TE2

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TE2<br/>(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 2 transfer error flag.                                                    |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>          |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>      |

Reference Manual to  
LL API cross  
reference:

### LL\_DMA\_IsActiveFlag\_TE3

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TE3<br/>(DMA_TypeDef * DMAx)</code> |
| Function description | Get Channel 3 transfer error flag.                                                    |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul>          |
| Return values        | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>      |

- Reference Manual to • ISR TEIF3 LL\_DMA\_IsActiveFlag\_TE3  
LL API cross reference:

### **LL\_DMA\_IsActiveFlag\_TE4**

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TE4(DMA_TypeDef * DMAx)</b> |
| Function description | Get Channel 4 transfer error flag.                                         |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                               |
| Return values        | • <b>State:</b> of bit (1 or 0).                                           |

Reference Manual to • ISR TEIF4 LL\_DMA\_IsActiveFlag\_TE4  
LL API cross reference:

### **LL\_DMA\_IsActiveFlag\_TE5**

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TE5(DMA_TypeDef * DMAx)</b> |
| Function description | Get Channel 5 transfer error flag.                                         |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                               |
| Return values        | • <b>State:</b> of bit (1 or 0).                                           |

Reference Manual to • ISR TEIF5 LL\_DMA\_IsActiveFlag\_TE5  
LL API cross reference:

### **LL\_DMA\_IsActiveFlag\_TE6**

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TE6(DMA_TypeDef * DMAx)</b> |
| Function description | Get Channel 6 transfer error flag.                                         |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                               |
| Return values        | • <b>State:</b> of bit (1 or 0).                                           |

Reference Manual to • ISR TEIF6 LL\_DMA\_IsActiveFlag\_TE6  
LL API cross reference:

### **LL\_DMA\_IsActiveFlag\_TE7**

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE uint32_t LL_DMA_IsActiveFlag_TE7(DMA_TypeDef * DMAx)</b> |
| Function description | Get Channel 7 transfer error flag.                                         |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                               |
| Return values        | • <b>State:</b> of bit (1 or 0).                                           |

- Reference Manual to • ISR TEIF7 LL\_DMA\_IsActiveFlag\_TE7  
LL API cross reference:

### LL\_DMA\_ClearFlag\_GI1

Function name **\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_GI1  
(DMA\_TypeDef \* DMAx)**

Function description Clear Channel 1 global interrupt flag.

Parameters • **DMAx:** DMAx Instance

Return values • **None**

- Reference Manual to • IFCR CGIF1 LL\_DMA\_ClearFlag\_GI1  
LL API cross reference:

### LL\_DMA\_ClearFlag\_GI2

Function name **\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_GI2  
(DMA\_TypeDef \* DMAx)**

Function description Clear Channel 2 global interrupt flag.

Parameters • **DMAx:** DMAx Instance

Return values • **None**

- Reference Manual to • IFCR CGIF2 LL\_DMA\_ClearFlag\_GI2  
LL API cross reference:

### LL\_DMA\_ClearFlag\_GI3

Function name **\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_GI3  
(DMA\_TypeDef \* DMAx)**

Function description Clear Channel 3 global interrupt flag.

Parameters • **DMAx:** DMAx Instance

Return values • **None**

- Reference Manual to • IFCR CGIF3 LL\_DMA\_ClearFlag\_GI3  
LL API cross reference:

### LL\_DMA\_ClearFlag\_GI4

Function name **\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_GI4  
(DMA\_TypeDef \* DMAx)**

Function description Clear Channel 4 global interrupt flag.

Parameters • **DMAx:** DMAx Instance

Return values • **None**

- Reference Manual to IFCR CGIF4 LL\_DMA\_ClearFlag\_GI4  
LL API cross reference:
- IFCR CGIF4 LL\_DMA\_ClearFlag\_GI4

### **LL\_DMA\_ClearFlag\_GI5**

Function name **\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_GI5(DMA\_TypeDef \* DMAx)**

Function description Clear Channel 5 global interrupt flag.

Parameters • **DMAx:** DMAx Instance

Return values • **None**

- Reference Manual to IFCR CGIF5 LL\_DMA\_ClearFlag\_GI5  
LL API cross reference:
- IFCR CGIF5 LL\_DMA\_ClearFlag\_GI5

### **LL\_DMA\_ClearFlag\_GI6**

Function name **\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_GI6(DMA\_TypeDef \* DMAx)**

Function description Clear Channel 6 global interrupt flag.

Parameters • **DMAx:** DMAx Instance

Return values • **None**

- Reference Manual to IFCR CGIF6 LL\_DMA\_ClearFlag\_GI6  
LL API cross reference:
- IFCR CGIF6 LL\_DMA\_ClearFlag\_GI6

### **LL\_DMA\_ClearFlag\_GI7**

Function name **\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_GI7(DMA\_TypeDef \* DMAx)**

Function description Clear Channel 7 global interrupt flag.

Parameters • **DMAx:** DMAx Instance

Return values • **None**

- Reference Manual to IFCR CGIF7 LL\_DMA\_ClearFlag\_GI7  
LL API cross reference:
- IFCR CGIF7 LL\_DMA\_ClearFlag\_GI7

### **LL\_DMA\_ClearFlag\_TC1**

Function name **\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_TC1(DMA\_TypeDef \* DMAx)**

Function description Clear Channel 1 transfer complete flag.

Parameters • **DMAx:** DMAx Instance

Return values • **None**

- Reference Manual to IFCR CTCIF1 LL\_DMA\_ClearFlag\_TC1  
LL API cross reference:

### LL\_DMA\_ClearFlag\_TC2

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_ClearFlag_TC2(DMA_TypeDef * DMAx)</code> |
| Function description | Clear Channel 2 transfer complete flag.                                    |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                               |
| Return values        | • <b>None</b>                                                              |

Reference Manual to IFCR CTCIF2 LL\_DMA\_ClearFlag\_TC2  
LL API cross reference:

### LL\_DMA\_ClearFlag\_TC3

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_ClearFlag_TC3(DMA_TypeDef * DMAx)</code> |
| Function description | Clear Channel 3 transfer complete flag.                                    |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                               |
| Return values        | • <b>None</b>                                                              |

Reference Manual to IFCR CTCIF3 LL\_DMA\_ClearFlag\_TC3  
LL API cross reference:

### LL\_DMA\_ClearFlag\_TC4

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_ClearFlag_TC4(DMA_TypeDef * DMAx)</code> |
| Function description | Clear Channel 4 transfer complete flag.                                    |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                               |
| Return values        | • <b>None</b>                                                              |

Reference Manual to IFCR CTCIF4 LL\_DMA\_ClearFlag\_TC4  
LL API cross reference:

### LL\_DMA\_ClearFlag\_TC5

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_ClearFlag_TC5(DMA_TypeDef * DMAx)</code> |
| Function description | Clear Channel 5 transfer complete flag.                                    |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                               |
| Return values        | • <b>None</b>                                                              |

- Reference Manual to IFCR CTCIF5 LL\_DMA\_ClearFlag\_TC5  
LL API cross reference:
- IFCR CTCIF5 LL\_DMA\_ClearFlag\_TC5

### **LL\_DMA\_ClearFlag\_TC6**

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_DMA_ClearFlag_TC6(DMA_TypeDef * DMAx)</code></b> |
| Function description | Clear Channel 6 transfer complete flag.                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> </ul>   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                  |

Reference Manual to IFCR CTCIF6 LL\_DMA\_ClearFlag\_TC6  
LL API cross reference:

### **LL\_DMA\_ClearFlag\_TC7**

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_DMA_ClearFlag_TC7(DMA_TypeDef * DMAx)</code></b> |
| Function description | Clear Channel 7 transfer complete flag.                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> </ul>   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                  |

Reference Manual to IFCR CTCIF7 LL\_DMA\_ClearFlag\_TC7  
LL API cross reference:

### **LL\_DMA\_ClearFlag\_HT1**

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_DMA_ClearFlag_HT1(DMA_TypeDef * DMAx)</code></b> |
| Function description | Clear Channel 1 half transfer flag.                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> </ul>   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                  |

Reference Manual to IFCR CHTIF1 LL\_DMA\_ClearFlag\_HT1  
LL API cross reference:

### **LL\_DMA\_ClearFlag\_HT2**

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_DMA_ClearFlag_HT2(DMA_TypeDef * DMAx)</code></b> |
| Function description | Clear Channel 2 half transfer flag.                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> </ul>   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                  |

- Reference Manual to IFCR CHTIF2 LL\_DMA\_ClearFlag\_HT2  
LL API cross reference:

### LL\_DMA\_ClearFlag\_HT3

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_ClearFlag_HT3(DMA_TypeDef * DMAx)</code>   |
| Function description | Clear Channel 3 half transfer flag.                                          |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                |

Reference Manual to IFCR CHTIF3 LL\_DMA\_ClearFlag\_HT3  
LL API cross reference:

### LL\_DMA\_ClearFlag\_HT4

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_ClearFlag_HT4(DMA_TypeDef * DMAx)</code>   |
| Function description | Clear Channel 4 half transfer flag.                                          |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                |

Reference Manual to IFCR CHTIF4 LL\_DMA\_ClearFlag\_HT4  
LL API cross reference:

### LL\_DMA\_ClearFlag\_HT5

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_ClearFlag_HT5(DMA_TypeDef * DMAx)</code>   |
| Function description | Clear Channel 5 half transfer flag.                                          |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                |

Reference Manual to IFCR CHTIF5 LL\_DMA\_ClearFlag\_HT5  
LL API cross reference:

### LL\_DMA\_ClearFlag\_HT6

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DMA_ClearFlag_HT6(DMA_TypeDef * DMAx)</code>   |
| Function description | Clear Channel 6 half transfer flag.                                          |
| Parameters           | <ul style="list-style-type: none"><li>• <b>DMAx:</b> DMAx Instance</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                |

- Reference Manual to  
LL API cross  
reference:
- IFCR CHTIF6 LL\_DMA\_ClearFlag\_HT6

### **LL\_DMA\_ClearFlag\_HT7**

Function name      **\_\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_HT7  
(DMA\_TypeDef \* DMAx)**

Function description      Clear Channel 7 half transfer flag.

Parameters      • **DMAx:** DMAx Instance

Return values      • **None**

- Reference Manual to  
LL API cross  
reference:
- IFCR CHTIF7 LL\_DMA\_ClearFlag\_HT7

### **LL\_DMA\_ClearFlag\_TE1**

Function name      **\_\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_TE1  
(DMA\_TypeDef \* DMAx)**

Function description      Clear Channel 1 transfer error flag.

Parameters      • **DMAx:** DMAx Instance

Return values      • **None**

- Reference Manual to  
LL API cross  
reference:
- IFCR CTEIF1 LL\_DMA\_ClearFlag\_TE1

### **LL\_DMA\_ClearFlag\_TE2**

Function name      **\_\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_TE2  
(DMA\_TypeDef \* DMAx)**

Function description      Clear Channel 2 transfer error flag.

Parameters      • **DMAx:** DMAx Instance

Return values      • **None**

- Reference Manual to  
LL API cross  
reference:
- IFCR CTEIF2 LL\_DMA\_ClearFlag\_TE2

### **LL\_DMA\_ClearFlag\_TE3**

Function name      **\_\_STATIC\_INLINE void LL\_DMA\_ClearFlag\_TE3  
(DMA\_TypeDef \* DMAx)**

Function description      Clear Channel 3 transfer error flag.

Parameters      • **DMAx:** DMAx Instance

Return values      • **None**

- Reference Manual to IFCR CTEIF3 LL\_DMA\_ClearFlag\_TE3  
LL API cross reference:

### LL\_DMA\_ClearFlag\_TE4

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_DMA_ClearFlag_TE4(DMA_TypeDef * DMAx)</code> |
| Function description | Clear Channel 4 transfer error flag.                                      |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                              |
| Return values        | • <b>None</b>                                                             |

Reference Manual to IFCR CTEIF4 LL\_DMA\_ClearFlag\_TE4  
LL API cross reference:

### LL\_DMA\_ClearFlag\_TE5

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_DMA_ClearFlag_TE5(DMA_TypeDef * DMAx)</code> |
| Function description | Clear Channel 5 transfer error flag.                                      |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                              |
| Return values        | • <b>None</b>                                                             |

Reference Manual to IFCR CTEIF5 LL\_DMA\_ClearFlag\_TE5  
LL API cross reference:

### LL\_DMA\_ClearFlag\_TE6

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_DMA_ClearFlag_TE6(DMA_TypeDef * DMAx)</code> |
| Function description | Clear Channel 6 transfer error flag.                                      |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                              |
| Return values        | • <b>None</b>                                                             |

Reference Manual to IFCR CTEIF6 LL\_DMA\_ClearFlag\_TE6  
LL API cross reference:

### LL\_DMA\_ClearFlag\_TE7

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_DMA_ClearFlag_TE7(DMA_TypeDef * DMAx)</code> |
| Function description | Clear Channel 7 transfer error flag.                                      |
| Parameters           | • <b>DMAx:</b> DMAx Instance                                              |
| Return values        | • <b>None</b>                                                             |

Reference Manual to  
LL API cross  
reference:

- IFCR CTEIF7 LL\_DMA\_ClearFlag\_TE7

### **LL\_DMA\_EnableIT\_TC**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_DMA_EnableIT_TC (DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                 |
| Function description                              | Enable Transfer complete interrupt.                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR TCIE LL_DMA_EnableIT_TC</li> </ul>                                                                                                                                                                                                                                                                                                                            |

### **LL\_DMA\_EnableIT\_HT**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_DMA_EnableIT_HT (DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                 |
| Function description                              | Enable Half transfer interrupt.                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_DMA_CHANNEL_1</li> <li>- LL_DMA_CHANNEL_2</li> <li>- LL_DMA_CHANNEL_3</li> <li>- LL_DMA_CHANNEL_4</li> <li>- LL_DMA_CHANNEL_5</li> <li>- LL_DMA_CHANNEL_6</li> <li>- LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR HTIE LL_DMA_EnableIT_HT</li> </ul>                                                                                                                                                                                                                                                                                                                            |

### **LL\_DMA\_EnableIT\_TE**

|                      |                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_DMA_EnableIT_TE (DMA_TypeDef * DMAx, uint32_t Channel)</code> |
| Function description | Enable Transfer error interrupt.                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> </ul>             |

- **Channel:** This parameter can be one of the following values:
  - LL\_DMA\_CHANNEL\_1
  - LL\_DMA\_CHANNEL\_2
  - LL\_DMA\_CHANNEL\_3
  - LL\_DMA\_CHANNEL\_4
  - LL\_DMA\_CHANNEL\_5
  - LL\_DMA\_CHANNEL\_6
  - LL\_DMA\_CHANNEL\_7

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CCR TEIE LL\_DMA\_EnableIT\_TE

**LL\_DMA\_DisableIT\_TC**

Function name      **`_STATIC_INLINE void LL_DMA_DisableIT_TC (DMA_TypeDef * DMAx, uint32_t Channel)`**

Function description      Disable Transfer complete interrupt.

Parameters

- **DMAx:** DMAx Instance
- **Channel:** This parameter can be one of the following values:
  - LL\_DMA\_CHANNEL\_1
  - LL\_DMA\_CHANNEL\_2
  - LL\_DMA\_CHANNEL\_3
  - LL\_DMA\_CHANNEL\_4
  - LL\_DMA\_CHANNEL\_5
  - LL\_DMA\_CHANNEL\_6
  - LL\_DMA\_CHANNEL\_7

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CCR TCIE LL\_DMA\_DisableIT\_TC

**LL\_DMA\_DisableIT\_HT**

Function name      **`_STATIC_INLINE void LL_DMA_DisableIT_HT (DMA_TypeDef * DMAx, uint32_t Channel)`**

Function description      Disable Half transfer interrupt.

Parameters

- **DMAx:** DMAx Instance
- **Channel:** This parameter can be one of the following values:
  - LL\_DMA\_CHANNEL\_1
  - LL\_DMA\_CHANNEL\_2
  - LL\_DMA\_CHANNEL\_3
  - LL\_DMA\_CHANNEL\_4
  - LL\_DMA\_CHANNEL\_5
  - LL\_DMA\_CHANNEL\_6
  - LL\_DMA\_CHANNEL\_7

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CCR HTIE LL\_DMA\_DisableIT\_HT

### **LL\_DMA\_DisableIT\_TE**

Function name

`_STATIC_INLINE void LL_DMA_DisableIT_TE (DMA_TypeDef * DMAx, uint32_t Channel)`

Function description

Disable Transfer error interrupt.

Parameters

- **DMAx:** DMAx Instance
- **Channel:** This parameter can be one of the following values:
  - LL\_DMA\_CHANNEL\_1
  - LL\_DMA\_CHANNEL\_2
  - LL\_DMA\_CHANNEL\_3
  - LL\_DMA\_CHANNEL\_4
  - LL\_DMA\_CHANNEL\_5
  - LL\_DMA\_CHANNEL\_6
  - LL\_DMA\_CHANNEL\_7

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CCR TEIE LL\_DMA\_DisableIT\_TE

### **LL\_DMA\_IsEnabledIT\_TC**

Function name

`_STATIC_INLINE uint32_t LL_DMA_IsEnabledIT_TC (DMA_TypeDef * DMAx, uint32_t Channel)`

Function description

Check if Transfer complete Interrupt is enabled.

Parameters

- **DMAx:** DMAx Instance
- **Channel:** This parameter can be one of the following values:
  - LL\_DMA\_CHANNEL\_1
  - LL\_DMA\_CHANNEL\_2
  - LL\_DMA\_CHANNEL\_3
  - LL\_DMA\_CHANNEL\_4
  - LL\_DMA\_CHANNEL\_5
  - LL\_DMA\_CHANNEL\_6
  - LL\_DMA\_CHANNEL\_7

Return values

- **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- CCR TCIE LL\_DMA\_IsEnabledIT\_TC

### **LL\_DMA\_IsEnabledIT\_HT**

Function name

`_STATIC_INLINE uint32_t LL_DMA_IsEnabledIT_HT (DMA_TypeDef * DMAx, uint32_t Channel)`

Function description

Check if Half transfer Interrupt is enabled.

Parameters

- **DMAx:** DMAx Instance

- **Channel:** This parameter can be one of the following values:
  - LL\_DMA\_CHANNEL\_1
  - LL\_DMA\_CHANNEL\_2
  - LL\_DMA\_CHANNEL\_3
  - LL\_DMA\_CHANNEL\_4
  - LL\_DMA\_CHANNEL\_5
  - LL\_DMA\_CHANNEL\_6
  - LL\_DMA\_CHANNEL\_7

Return values

- **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- CCR HTIE LL\_DMA\_IsEnabledIT\_HT

**LL\_DMA\_IsEnabledIT\_TE**

Function name **`__STATIC_INLINE uint32_t LL_DMA_IsEnabledIT_TE(DMA_TypeDef * DMax, uint32_t Channel)`**

Function description

Check if Transfer error Interrupt is enabled.

Parameters

- **DMax:** DMAx Instance
- **Channel:** This parameter can be one of the following values:
  - LL\_DMA\_CHANNEL\_1
  - LL\_DMA\_CHANNEL\_2
  - LL\_DMA\_CHANNEL\_3
  - LL\_DMA\_CHANNEL\_4
  - LL\_DMA\_CHANNEL\_5
  - LL\_DMA\_CHANNEL\_6
  - LL\_DMA\_CHANNEL\_7

Return values

- **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- CCR TEIE LL\_DMA\_IsEnabledIT\_TE

**LL\_DMA\_Init**

Function name **`uint32_t LL_DMA_Init (DMA_TypeDef * DMax, uint32_t Channel, LL_DMA_InitTypeDef * DMA_InitStruct)`**

Function description

Initialize the DMA registers according to the specified parameters in DMA\_InitStruct.

Parameters

- **DMax:** DMAx Instance
- **Channel:** This parameter can be one of the following values:
  - LL\_DMA\_CHANNEL\_1
  - LL\_DMA\_CHANNEL\_2
  - LL\_DMA\_CHANNEL\_3
  - LL\_DMA\_CHANNEL\_4
  - LL\_DMA\_CHANNEL\_5
  - LL\_DMA\_CHANNEL\_6
  - LL\_DMA\_CHANNEL\_7
- **DMA\_InitStruct:** pointer to a LL\_DMA\_InitTypeDef structure.

---

|               |                                                                                                                                                                                                                                      |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: DMA registers are initialized</li> <li>– ERROR: Not applicable</li> </ul> </li> </ul> |
| Notes         | <ul style="list-style-type: none"> <li>• To convert DMAx_ChannelInstance to DMAx Instance and Channel, use helper macros : __LL_DMA_GET_INSTANCE __LL_DMA_GET_CHANNEL</li> </ul>                                                     |

### LL\_DMA\_DeInit

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t LL_DMA_DeInit (DMA_TypeDef * DMAx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                                           |
| Function description | De-initialize the DMA registers to their default reset values.                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMAx:</b> DMAx Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_DMA_CHANNEL_1</li> <li>– LL_DMA_CHANNEL_2</li> <li>– LL_DMA_CHANNEL_3</li> <li>– LL_DMA_CHANNEL_4</li> <li>– LL_DMA_CHANNEL_5</li> <li>– LL_DMA_CHANNEL_6</li> <li>– LL_DMA_CHANNEL_7</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: DMA registers are de-initialized</li> <li>– ERROR: DMA registers are not de-initialized</li> </ul> </li> </ul>                                                                                                                                                        |

### LL\_DMA\_StructInit

|                      |                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void LL_DMA_StructInit (LL_DMA_InitTypeDef * DMA_InitStruct)</code>                                             |
| Function description | Set each LL_DMA_InitTypeDef field to default value.                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>DMA_InitStruct:</b> Pointer to a LL_DMA_InitTypeDef structure.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                       |

## 65.3 DMA Firmware driver defines

### 65.3.1 DMA

#### CHANNEL

|                               |               |
|-------------------------------|---------------|
| <code>LL_DMA_CHANNEL_1</code> | DMA Channel 1 |
| <code>LL_DMA_CHANNEL_2</code> | DMA Channel 2 |
| <code>LL_DMA_CHANNEL_3</code> | DMA Channel 3 |
| <code>LL_DMA_CHANNEL_4</code> | DMA Channel 4 |
| <code>LL_DMA_CHANNEL_5</code> | DMA Channel 5 |
| <code>LL_DMA_CHANNEL_6</code> | DMA Channel 6 |
| <code>LL_DMA_CHANNEL_7</code> | DMA Channel 7 |

`LL_DMA_CHANNEL_ALL` DMA Channel all (used only for function)

***Clear Flags Defines***

|                                 |                                  |
|---------------------------------|----------------------------------|
| <code>LL_DMA_IFCR_CGIF1</code>  | Channel 1 global flag            |
| <code>LL_DMA_IFCR_CTCIF1</code> | Channel 1 transfer complete flag |
| <code>LL_DMA_IFCR_CHTIF1</code> | Channel 1 half transfer flag     |
| <code>LL_DMA_IFCR_CTEIF1</code> | Channel 1 transfer error flag    |
| <code>LL_DMA_IFCR_CGIF2</code>  | Channel 2 global flag            |
| <code>LL_DMA_IFCR_CTCIF2</code> | Channel 2 transfer complete flag |
| <code>LL_DMA_IFCR_CHTIF2</code> | Channel 2 half transfer flag     |
| <code>LL_DMA_IFCR_CTEIF2</code> | Channel 2 transfer error flag    |
| <code>LL_DMA_IFCR_CGIF3</code>  | Channel 3 global flag            |
| <code>LL_DMA_IFCR_CTCIF3</code> | Channel 3 transfer complete flag |
| <code>LL_DMA_IFCR_CHTIF3</code> | Channel 3 half transfer flag     |
| <code>LL_DMA_IFCR_CTEIF3</code> | Channel 3 transfer error flag    |
| <code>LL_DMA_IFCR_CGIF4</code>  | Channel 4 global flag            |
| <code>LL_DMA_IFCR_CTCIF4</code> | Channel 4 transfer complete flag |
| <code>LL_DMA_IFCR_CHTIF4</code> | Channel 4 half transfer flag     |
| <code>LL_DMA_IFCR_CTEIF4</code> | Channel 4 transfer error flag    |
| <code>LL_DMA_IFCR_CGIF5</code>  | Channel 5 global flag            |
| <code>LL_DMA_IFCR_CTCIF5</code> | Channel 5 transfer complete flag |
| <code>LL_DMA_IFCR_CHTIF5</code> | Channel 5 half transfer flag     |
| <code>LL_DMA_IFCR_CTEIF5</code> | Channel 5 transfer error flag    |
| <code>LL_DMA_IFCR_CGIF6</code>  | Channel 6 global flag            |
| <code>LL_DMA_IFCR_CTCIF6</code> | Channel 6 transfer complete flag |
| <code>LL_DMA_IFCR_CHTIF6</code> | Channel 6 half transfer flag     |
| <code>LL_DMA_IFCR_CTEIF6</code> | Channel 6 transfer error flag    |
| <code>LL_DMA_IFCR_CGIF7</code>  | Channel 7 global flag            |
| <code>LL_DMA_IFCR_CTCIF7</code> | Channel 7 transfer complete flag |
| <code>LL_DMA_IFCR_CHTIF7</code> | Channel 7 half transfer flag     |
| <code>LL_DMA_IFCR_CTEIF7</code> | Channel 7 transfer error flag    |

***Transfer Direction***

|                                                |                                |
|------------------------------------------------|--------------------------------|
| <code>LL_DMA_DIRECTION_PERIPH_TO_MEMORY</code> | Peripheral to memory direction |
| <code>LL_DMA_DIRECTION_MEMORY_TO_PERIPH</code> | Memory to peripheral direction |
| <code>LL_DMA_DIRECTION_MEMORY_TO_MEMORY</code> | Memory to memory direction     |

***Get Flags Defines***

|                              |                       |
|------------------------------|-----------------------|
| <code>LL_DMA_ISR_GIF1</code> | Channel 1 global flag |
|------------------------------|-----------------------|

|                               |                                  |
|-------------------------------|----------------------------------|
| <code>LL_DMA_ISR_TCIF1</code> | Channel 1 transfer complete flag |
| <code>LL_DMA_ISR_HTIF1</code> | Channel 1 half transfer flag     |
| <code>LL_DMA_ISR_TEIF1</code> | Channel 1 transfer error flag    |
| <code>LL_DMA_ISR_GIF2</code>  | Channel 2 global flag            |
| <code>LL_DMA_ISR_TCIF2</code> | Channel 2 transfer complete flag |
| <code>LL_DMA_ISR_HTIF2</code> | Channel 2 half transfer flag     |
| <code>LL_DMA_ISR_TEIF2</code> | Channel 2 transfer error flag    |
| <code>LL_DMA_ISR_GIF3</code>  | Channel 3 global flag            |
| <code>LL_DMA_ISR_TCIF3</code> | Channel 3 transfer complete flag |
| <code>LL_DMA_ISR_HTIF3</code> | Channel 3 half transfer flag     |
| <code>LL_DMA_ISR_TEIF3</code> | Channel 3 transfer error flag    |
| <code>LL_DMA_ISR_GIF4</code>  | Channel 4 global flag            |
| <code>LL_DMA_ISR_TCIF4</code> | Channel 4 transfer complete flag |
| <code>LL_DMA_ISR_HTIF4</code> | Channel 4 half transfer flag     |
| <code>LL_DMA_ISR_TEIF4</code> | Channel 4 transfer error flag    |
| <code>LL_DMA_ISR_GIF5</code>  | Channel 5 global flag            |
| <code>LL_DMA_ISR_TCIF5</code> | Channel 5 transfer complete flag |
| <code>LL_DMA_ISR_HTIF5</code> | Channel 5 half transfer flag     |
| <code>LL_DMA_ISR_TEIF5</code> | Channel 5 transfer error flag    |
| <code>LL_DMA_ISR_GIF6</code>  | Channel 6 global flag            |
| <code>LL_DMA_ISR_TCIF6</code> | Channel 6 transfer complete flag |
| <code>LL_DMA_ISR_HTIF6</code> | Channel 6 half transfer flag     |
| <code>LL_DMA_ISR_TEIF6</code> | Channel 6 transfer error flag    |
| <code>LL_DMA_ISR_GIF7</code>  | Channel 7 global flag            |
| <code>LL_DMA_ISR_TCIF7</code> | Channel 7 transfer complete flag |
| <code>LL_DMA_ISR_HTIF7</code> | Channel 7 half transfer flag     |
| <code>LL_DMA_ISR_TEIF7</code> | Channel 7 transfer error flag    |

***IT Defines***

|                              |                             |
|------------------------------|-----------------------------|
| <code>LL_DMA_CCR_TCIE</code> | Transfer complete interrupt |
| <code>LL_DMA_CCR_HTIE</code> | Half Transfer interrupt     |
| <code>LL_DMA_CCR_TEIE</code> | Transfer error interrupt    |

***Memory data alignment***

|                                         |                                  |
|-----------------------------------------|----------------------------------|
| <code>LL_DMA_MDATAALIGN_BYTE</code>     | Memory data alignment : Byte     |
| <code>LL_DMA_MDATAALIGN_HALFWORD</code> | Memory data alignment : HalfWord |
| <code>LL_DMA_MDATAALIGN_WORD</code>     | Memory data alignment : Word     |

***Memory increment mode***

`LL_DMA_MEMORY_INCREMENT` Memory increment mode Enable

`LL_DMA_MEMORY_NOINCREMENT` Memory increment mode Disable

***Transfer mode***

`LL_DMA_MODE_NORMAL` Normal Mode

`LL_DMA_MODE_CIRCULAR` Circular Mode

***Peripheral data alignment***

`LL_DMA_PDATAALIGN_BYTE` Peripheral data alignment : Byte

`LL_DMA_PDATAALIGN_HALFWORD` Peripheral data alignment : HalfWord

`LL_DMA_PDATAALIGN_WORD` Peripheral data alignment : Word

***Peripheral increment mode***

`LL_DMA_PERIPH_INCREMENT` Peripheral increment mode Enable

`LL_DMA_PERIPH_NOINCREMENT` Peripheral increment mode Disable

***Transfer Priority level***

`LL_DMA_PRIORITY_LOW` Priority level : Low

`LL_DMA_PRIORITY_MEDIUM` Priority level : Medium

`LL_DMA_PRIORITY_HIGH` Priority level : High

`LL_DMA_PRIORITY_VERYHIGH` Priority level : Very\_High

***Convert DMAxChannely***

- `__LL_DMA_GET_INSTANCE` **Description:**
- Convert DMAx\_Channely into DMAx.
- Parameters:**
- `__CHANNEL_INSTANCE__`: DMAx\_Channely

**Return value:**

- DMAx

- `__LL_DMA_GET_CHANNEL` **Description:**
- Convert DMAx\_Channely into `LL_DMA_CHANNEL_y`.

**Parameters:**

- `__CHANNEL_INSTANCE__`: DMAx\_Channely

**Return value:**

- `LL_DMA_CHANNEL_y`

- `__LL_DMA_GET_CHANNEL_INSTANCE` **Description:**
- Convert DMA Instance DMAx and `LL_DMA_CHANNEL_y` into DMAx\_Channely.

**Parameters:**

- `__DMA_INSTANCE__`: DMAx
- `__CHANNEL__`: LL\_DMA\_CHANNEL\_y

**Return value:**

- DMAx\_Channely

**Common Write and read registers macros****LL\_DMA\_WriteReg****Description:**

- Write a value in DMA register.

**Parameters:**

- `__INSTANCE__`: DMA Instance
- `__REG__`: Register to be written
- `__VALUE__`: Value to be written in the register

**Return value:**

- None

**LL\_DMA\_ReadReg****Description:**

- Read a value in DMA register.

**Parameters:**

- `__INSTANCE__`: DMA Instance
- `__REG__`: Register to be read

**Return value:**

- Register: value

## 66 LL EXTI Generic Driver

### 66.1 EXTI Firmware driver registers structures

#### 66.1.1 LL\_EXTI\_InitTypeDef

##### Data Fields

- *uint32\_t Line\_0\_31*
- *uint32\_t Line\_32\_63*
- *FunctionalState LineCommand*
- *uint8\_t Mode*
- *uint8\_t Trigger*

##### Field Documentation

- ***uint32\_t LL\_EXTI\_InitTypeDef::Line\_0\_31***  
Specifies the EXTI lines to be enabled or disabled for Lines in range 0 to 31 This parameter can be any combination of [\*EXTI\\_LL\\_EC\\_LINE\*](#)
- ***uint32\_t LL\_EXTI\_InitTypeDef::Line\_32\_63***  
Specifies the EXTI lines to be enabled or disabled for Lines in range 32 to 63 This parameter can be any combination of [\*EXTI\\_LL\\_EC\\_LINE\*](#)
- ***FunctionalState LL\_EXTI\_InitTypeDef::LineCommand***  
Specifies the new state of the selected EXTI lines. This parameter can be set either to ENABLE or DISABLE
- ***uint8\_t LL\_EXTI\_InitTypeDef::Mode***  
Specifies the mode for the EXTI lines. This parameter can be a value of [\*EXTI\\_LL\\_EC\\_MODE\*](#).
- ***uint8\_t LL\_EXTI\_InitTypeDef::Trigger***  
Specifies the trigger signal active edge for the EXTI lines. This parameter can be a value of [\*EXTI\\_LL\\_EC\\_TRIGGER\*](#).

### 66.2 EXTI Firmware driver API description

#### 66.2.1 Detailed description of functions

##### LL\_EXTI\_EnableIT\_0\_31

Function name      `__STATIC_INLINE void LL_EXTI_EnableIT_0_31 (uint32_t ExtiLine)`

Function description      Enable ExtiLine Interrupt request for Lines in range 0 to 31.

- Parameters
- **ExtiLine:** This parameter can be one of the following values:
    - `LL_EXTI_LINE_0`
    - `LL_EXTI_LINE_1`
    - `LL_EXTI_LINE_2`
    - `LL_EXTI_LINE_3`
    - `LL_EXTI_LINE_4`
    - `LL_EXTI_LINE_5`
    - `LL_EXTI_LINE_6`
    - `LL_EXTI_LINE_7`
    - `LL_EXTI_LINE_8`
    - `LL_EXTI_LINE_9`

- LL\_EXTI\_LINE\_10
- LL\_EXTI\_LINE\_11
- LL\_EXTI\_LINE\_12
- LL\_EXTI\_LINE\_13
- LL\_EXTI\_LINE\_14
- LL\_EXTI\_LINE\_15
- LL\_EXTI\_LINE\_16
- LL\_EXTI\_LINE\_17
- LL\_EXTI\_LINE\_18
- LL\_EXTI\_LINE\_19
- LL\_EXTI\_LINE\_20
- LL\_EXTI\_LINE\_21
- LL\_EXTI\_LINE\_22
- LL\_EXTI\_LINE\_23
- LL\_EXTI\_LINE\_24
- LL\_EXTI\_LINE\_25
- LL\_EXTI\_LINE\_26
- LL\_EXTI\_LINE\_27
- LL\_EXTI\_LINE\_28
- LL\_EXTI\_LINE\_29
- LL\_EXTI\_LINE\_30
- LL\_EXTI\_LINE\_31
- LL\_EXTI\_LINE\_ALL\_0\_31

Return values

- **None**

Notes

- The reset value for the direct or internal lines (see RM) is set to 1 in order to enable the interrupt by default. Bits are set automatically at Power on.
- Please check each device line mapping for EXTI Line availability

Reference Manual to  
LL API cross  
reference:

- IMR IMx LL\_EXTI\_EnableIT\_0\_31

### **LL\_EXTI\_EnableIT\_32\_63**

Function name      **\_\_STATIC\_INLINE void LL\_EXTI\_EnableIT\_32\_63 (uint32\_t ExtiLine)**

Function description      Enable ExtiLine Interrupt request for Lines in range 32 to 63.

Parameters

- **ExtiLine:** This parameter can be one of the following values:
  - LL\_EXTI\_LINE\_32
  - LL\_EXTI\_LINE\_33
  - LL\_EXTI\_LINE\_34
  - LL\_EXTI\_LINE\_35
  - LL\_EXTI\_LINE\_36
  - LL\_EXTI\_LINE\_37
  - LL\_EXTI\_LINE\_38
  - LL\_EXTI\_LINE\_39
  - LL\_EXTI\_LINE\_ALL\_32\_63

Return values

- **None**

|                                                   |                                                                                                                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                             | <ul style="list-style-type: none"> <li>The reset value for the direct lines (lines from 32 to 34, line 39) is set to 1 in order to enable the interrupt by default. Bits are set automatically at Power on.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>IMR2 IMx LL_EXTI_EnableIT_32_63</li> </ul>                                                                                                                                      |

### LL\_EXTI\_DisableIT\_0\_31

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_EXTI_DisableIT_0_31 (uint32_t ExtiLine)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Disable ExtiLine Interrupt request for Lines in range 0 to 31.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>ExtiLine:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_EXTI_LINE_0</code></li> <li>- <code>LL_EXTI_LINE_1</code></li> <li>- <code>LL_EXTI_LINE_2</code></li> <li>- <code>LL_EXTI_LINE_3</code></li> <li>- <code>LL_EXTI_LINE_4</code></li> <li>- <code>LL_EXTI_LINE_5</code></li> <li>- <code>LL_EXTI_LINE_6</code></li> <li>- <code>LL_EXTI_LINE_7</code></li> <li>- <code>LL_EXTI_LINE_8</code></li> <li>- <code>LL_EXTI_LINE_9</code></li> <li>- <code>LL_EXTI_LINE_10</code></li> <li>- <code>LL_EXTI_LINE_11</code></li> <li>- <code>LL_EXTI_LINE_12</code></li> <li>- <code>LL_EXTI_LINE_13</code></li> <li>- <code>LL_EXTI_LINE_14</code></li> <li>- <code>LL_EXTI_LINE_15</code></li> <li>- <code>LL_EXTI_LINE_16</code></li> <li>- <code>LL_EXTI_LINE_17</code></li> <li>- <code>LL_EXTI_LINE_18</code></li> <li>- <code>LL_EXTI_LINE_19</code></li> <li>- <code>LL_EXTI_LINE_20</code></li> <li>- <code>LL_EXTI_LINE_21</code></li> <li>- <code>LL_EXTI_LINE_22</code></li> <li>- <code>LL_EXTI_LINE_23</code></li> <li>- <code>LL_EXTI_LINE_24</code></li> <li>- <code>LL_EXTI_LINE_25</code></li> <li>- <code>LL_EXTI_LINE_26</code></li> <li>- <code>LL_EXTI_LINE_27</code></li> <li>- <code>LL_EXTI_LINE_28</code></li> <li>- <code>LL_EXTI_LINE_29</code></li> <li>- <code>LL_EXTI_LINE_30</code></li> <li>- <code>LL_EXTI_LINE_31</code></li> <li>- <code>LL_EXTI_LINE_ALL_0_31</code></li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>The reset value for the direct or internal lines (see RM) is set to 1 in order to enable the interrupt by default. Bits are set automatically at Power on.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

- Please check each device line mapping for EXTI Line availability
  - IMR IMx LL\_EXTI\_DisableIT\_0\_31
- Reference Manual to  
LL API cross  
reference:

### **LL\_EXTI\_DisableIT\_32\_63**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_EXTI_DisableIT_32_63 (uint32_t ExtiLine)</code></b>                                                                                                                                                                                                                                                                                                                                                  |
| Function description                              | Disable ExtiLine Interrupt request for Lines in range 32 to 63.                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_EXTI_LINE_32</li> <li>– LL_EXTI_LINE_33</li> <li>– LL_EXTI_LINE_34</li> <li>– LL_EXTI_LINE_35</li> <li>– LL_EXTI_LINE_36</li> <li>– LL_EXTI_LINE_37</li> <li>– LL_EXTI_LINE_38</li> <li>– LL_EXTI_LINE_39</li> <li>– LL_EXTI_LINE_ALL_32_63</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• The reset value for the direct lines (lines from 32 to 34, line 39) is set to 1 in order to enable the interrupt by default. Bits are set automatically at Power on.</li> </ul>                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IMR2 IMx LL_EXTI_DisableIT_32_63</li> </ul>                                                                                                                                                                                                                                                                                                                                                  |

### **LL\_EXTI\_IsEnabledIT\_0\_31**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_EXTI_IsEnabledIT_0_31 (uint32_t ExtiLine)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Indicate if ExtiLine Interrupt request is enabled for Lines in range 0 to 31.                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_EXTI_LINE_0</li> <li>– LL_EXTI_LINE_1</li> <li>– LL_EXTI_LINE_2</li> <li>– LL_EXTI_LINE_3</li> <li>– LL_EXTI_LINE_4</li> <li>– LL_EXTI_LINE_5</li> <li>– LL_EXTI_LINE_6</li> <li>– LL_EXTI_LINE_7</li> <li>– LL_EXTI_LINE_8</li> <li>– LL_EXTI_LINE_9</li> <li>– LL_EXTI_LINE_10</li> <li>– LL_EXTI_LINE_11</li> <li>– LL_EXTI_LINE_12</li> </ul> </li> </ul> |

- LL\_EXTI\_LINE\_13
- LL\_EXTI\_LINE\_14
- LL\_EXTI\_LINE\_15
- LL\_EXTI\_LINE\_16
- LL\_EXTI\_LINE\_17
- LL\_EXTI\_LINE\_18
- LL\_EXTI\_LINE\_19
- LL\_EXTI\_LINE\_20
- LL\_EXTI\_LINE\_21
- LL\_EXTI\_LINE\_22
- LL\_EXTI\_LINE\_23
- LL\_EXTI\_LINE\_24
- LL\_EXTI\_LINE\_25
- LL\_EXTI\_LINE\_26
- LL\_EXTI\_LINE\_27
- LL\_EXTI\_LINE\_28
- LL\_EXTI\_LINE\_29
- LL\_EXTI\_LINE\_30
- LL\_EXTI\_LINE\_31
- LL\_EXTI\_LINE\_ALL\_0\_31

Return values

- **State:** of bit (1 or 0).

Notes

- The reset value for the direct or internal lines (see RM) is set to 1 in order to enable the interrupt by default. Bits are set automatically at Power on.
- Please check each device line mapping for EXTI Line availability

Reference Manual to  
LL API cross  
reference:

- IMR IMx LL\_EXTI\_IsEnabledIT\_0\_31

### LL\_EXTI\_IsEnabledIT\_32\_63

Function name `__STATIC_INLINE uint32_t LL_EXTI_IsEnabledIT_32_63  
(uint32_t ExtiLine)`

Function description Indicate if ExtiLine Interrupt request is enabled for Lines in range 32 to 63.

Parameters

- **ExtiLine:** This parameter can be one of the following values:
  - LL\_EXTI\_LINE\_32
  - LL\_EXTI\_LINE\_33
  - LL\_EXTI\_LINE\_34
  - LL\_EXTI\_LINE\_35
  - LL\_EXTI\_LINE\_36
  - LL\_EXTI\_LINE\_37
  - LL\_EXTI\_LINE\_38
  - LL\_EXTI\_LINE\_39
  - LL\_EXTI\_LINE\_ALL\_32\_63

Return values

- **State:** of bit (1 or 0).

Notes

- The reset value for the direct lines (lines from 32 to 34, line 39) is set to 1 in order to enable the interrupt by default. Bits

are set automatically at Power on.

Reference Manual to  
LL API cross  
reference:

- IMR2 IMx LL\_EXTI\_IsEnabledIT\_32\_63

### **LL\_EXTI\_EnableEvent\_0\_31**

Function name **\_STATIC\_INLINE void LL\_EXTI\_EnableEvent\_0\_31 (uint32\_t ExtiLine)**

Function description Enable ExtiLine Event request for Lines in range 0 to 31.

Parameters • **ExtiLine:** This parameter can be one of the following values:

- LL\_EXTI\_LINE\_0
- LL\_EXTI\_LINE\_1
- LL\_EXTI\_LINE\_2
- LL\_EXTI\_LINE\_3
- LL\_EXTI\_LINE\_4
- LL\_EXTI\_LINE\_5
- LL\_EXTI\_LINE\_6
- LL\_EXTI\_LINE\_7
- LL\_EXTI\_LINE\_8
- LL\_EXTI\_LINE\_9
- LL\_EXTI\_LINE\_10
- LL\_EXTI\_LINE\_11
- LL\_EXTI\_LINE\_12
- LL\_EXTI\_LINE\_13
- LL\_EXTI\_LINE\_14
- LL\_EXTI\_LINE\_15
- LL\_EXTI\_LINE\_16
- LL\_EXTI\_LINE\_17
- LL\_EXTI\_LINE\_18
- LL\_EXTI\_LINE\_19
- LL\_EXTI\_LINE\_20
- LL\_EXTI\_LINE\_21
- LL\_EXTI\_LINE\_22
- LL\_EXTI\_LINE\_23
- LL\_EXTI\_LINE\_24
- LL\_EXTI\_LINE\_25
- LL\_EXTI\_LINE\_26
- LL\_EXTI\_LINE\_27
- LL\_EXTI\_LINE\_28
- LL\_EXTI\_LINE\_29
- LL\_EXTI\_LINE\_30
- LL\_EXTI\_LINE\_31
- LL\_EXTI\_LINE\_ALL\_0\_31

Return values • **None**

Notes • Please check each device line mapping for EXTI Line availability

Reference Manual to  
LL API cross  
reference:

- EMR EMx LL\_EXTI\_EnableEvent\_0\_31

### LL\_EXTI\_EnableEvent\_32\_63

Function name

**`__STATIC_INLINE void LL_EXTI_EnableEvent_32_63 (uint32_t ExtiLine)`**

Function description

Enable ExtiLine Event request for Lines in range 32 to 63.

Parameters

- **ExtiLine:** This parameter can be a combination of the following values:
  - LL\_EXTI\_LINE\_32
  - LL\_EXTI\_LINE\_33
  - LL\_EXTI\_LINE\_34
  - LL\_EXTI\_LINE\_35
  - LL\_EXTI\_LINE\_36
  - LL\_EXTI\_LINE\_37
  - LL\_EXTI\_LINE\_38
  - LL\_EXTI\_LINE\_39
  - LL\_EXTI\_LINE\_ALL\_32\_63

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- EMR2 EMx LL\_EXTI\_EnableEvent\_32\_63

### LL\_EXTI\_DisableEvent\_0\_31

Function name

**`__STATIC_INLINE void LL_EXTI_DisableEvent_0_31 (uint32_t ExtiLine)`**

Function description

Disable ExtiLine Event request for Lines in range 0 to 31.

Parameters

- **ExtiLine:** This parameter can be one of the following values:
  - LL\_EXTI\_LINE\_0
  - LL\_EXTI\_LINE\_1
  - LL\_EXTI\_LINE\_2
  - LL\_EXTI\_LINE\_3
  - LL\_EXTI\_LINE\_4
  - LL\_EXTI\_LINE\_5
  - LL\_EXTI\_LINE\_6
  - LL\_EXTI\_LINE\_7
  - LL\_EXTI\_LINE\_8
  - LL\_EXTI\_LINE\_9
  - LL\_EXTI\_LINE\_10
  - LL\_EXTI\_LINE\_11
  - LL\_EXTI\_LINE\_12
  - LL\_EXTI\_LINE\_13
  - LL\_EXTI\_LINE\_14
  - LL\_EXTI\_LINE\_15
  - LL\_EXTI\_LINE\_16
  - LL\_EXTI\_LINE\_17
  - LL\_EXTI\_LINE\_18

- LL\_EXTI\_LINE\_19
- LL\_EXTI\_LINE\_20
- LL\_EXTI\_LINE\_21
- LL\_EXTI\_LINE\_22
- LL\_EXTI\_LINE\_23
- LL\_EXTI\_LINE\_24
- LL\_EXTI\_LINE\_25
- LL\_EXTI\_LINE\_26
- LL\_EXTI\_LINE\_27
- LL\_EXTI\_LINE\_28
- LL\_EXTI\_LINE\_29
- LL\_EXTI\_LINE\_30
- LL\_EXTI\_LINE\_31
- LL\_EXTI\_LINE\_ALL\_0\_31

**Return values**

- **None**

**Notes**

- Please check each device line mapping for EXTI Line availability

**Reference Manual to**

LL API cross  
reference:

- EMR EMx LL\_EXTI\_DisableEvent\_0\_31

**LL\_EXTI\_DisableEvent\_32\_63****Function name**

**\_STATIC\_INLINE void LL\_EXTI\_DisableEvent\_32\_63  
(uint32\_t ExtiLine)**

**Function description**

Disable ExtiLine Event request for Lines in range 32 to 63.

**Parameters**

- **ExtiLine:** This parameter can be a combination of the following values:
  - LL\_EXTI\_LINE\_32
  - LL\_EXTI\_LINE\_33
  - LL\_EXTI\_LINE\_34
  - LL\_EXTI\_LINE\_35
  - LL\_EXTI\_LINE\_36
  - LL\_EXTI\_LINE\_37
  - LL\_EXTI\_LINE\_38
  - LL\_EXTI\_LINE\_39
  - LL\_EXTI\_LINE\_ALL\_32\_63

**Return values**

- **None**

**Reference Manual to**

LL API cross  
reference:

- EMR2 EMx LL\_EXTI\_DisableEvent\_32\_63

**LL\_EXTI\_IsEnabledEvent\_0\_31**

Function name      **\_STATIC\_INLINE uint32\_t LL\_EXTI\_IsEnabledEvent\_0\_31  
(uint32\_t ExtiLine)**

Function description      Indicate if ExtiLine Event request is enabled for Lines in range 0 to 31.

- Parameters
- **ExtiLine:** This parameter can be one of the following values:
    - LL\_EXTI\_LINE\_0
    - LL\_EXTI\_LINE\_1
    - LL\_EXTI\_LINE\_2
    - LL\_EXTI\_LINE\_3
    - LL\_EXTI\_LINE\_4
    - LL\_EXTI\_LINE\_5
    - LL\_EXTI\_LINE\_6
    - LL\_EXTI\_LINE\_7
    - LL\_EXTI\_LINE\_8
    - LL\_EXTI\_LINE\_9
    - LL\_EXTI\_LINE\_10
    - LL\_EXTI\_LINE\_11
    - LL\_EXTI\_LINE\_12
    - LL\_EXTI\_LINE\_13
    - LL\_EXTI\_LINE\_14
    - LL\_EXTI\_LINE\_15
    - LL\_EXTI\_LINE\_16
    - LL\_EXTI\_LINE\_17
    - LL\_EXTI\_LINE\_18
    - LL\_EXTI\_LINE\_19
    - LL\_EXTI\_LINE\_20
    - LL\_EXTI\_LINE\_21
    - LL\_EXTI\_LINE\_22
    - LL\_EXTI\_LINE\_23
    - LL\_EXTI\_LINE\_24
    - LL\_EXTI\_LINE\_25
    - LL\_EXTI\_LINE\_26
    - LL\_EXTI\_LINE\_27
    - LL\_EXTI\_LINE\_28
    - LL\_EXTI\_LINE\_29
    - LL\_EXTI\_LINE\_30
    - LL\_EXTI\_LINE\_31
    - LL\_EXTI\_LINE\_ALL\_0\_31

- Return values
- **State:** of bit (1 or 0).

- Notes
- Please check each device line mapping for EXTI Line availability

- Reference Manual to  
LL API cross  
reference:
- EMR EMx LL\_EXTI\_IsEnabledEvent\_0\_31

**LL\_EXTI\_IsEnabledEvent\_32\_63**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_EXTI_IsEnabledEvent_32_63<br/>(uint32_t ExtiLine)</code></b>                                                                                                                                                                                                                                                                                                                                      |
| Function description                              | Indicate if ExtiLine Event request is enabled for Lines in range 32 to 63.                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_32</li> <li>- LL_EXTI_LINE_33</li> <li>- LL_EXTI_LINE_34</li> <li>- LL_EXTI_LINE_35</li> <li>- LL_EXTI_LINE_36</li> <li>- LL_EXTI_LINE_37</li> <li>- LL_EXTI_LINE_38</li> <li>- LL_EXTI_LINE_39</li> <li>- LL_EXTI_LINE_ALL_32_63</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EMR2 EMx LL_EXTI_IsEnabledEvent_32_63</li> </ul>                                                                                                                                                                                                                                                                                                                                             |

**LL\_EXTI\_EnableRisingTrig\_0\_31**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_EXTI_EnableRisingTrig_0_31<br/>(uint32_t ExtiLine)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Enable ExtiLine Rising Edge Trigger for Lines in range 0 to 31.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_0</li> <li>- LL_EXTI_LINE_1</li> <li>- LL_EXTI_LINE_2</li> <li>- LL_EXTI_LINE_3</li> <li>- LL_EXTI_LINE_4</li> <li>- LL_EXTI_LINE_5</li> <li>- LL_EXTI_LINE_6</li> <li>- LL_EXTI_LINE_7</li> <li>- LL_EXTI_LINE_8</li> <li>- LL_EXTI_LINE_9</li> <li>- LL_EXTI_LINE_10</li> <li>- LL_EXTI_LINE_11</li> <li>- LL_EXTI_LINE_12</li> <li>- LL_EXTI_LINE_13</li> <li>- LL_EXTI_LINE_14</li> <li>- LL_EXTI_LINE_15</li> <li>- LL_EXTI_LINE_16</li> <li>- LL_EXTI_LINE_18</li> <li>- LL_EXTI_LINE_19</li> <li>- LL_EXTI_LINE_20</li> <li>- LL_EXTI_LINE_21</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_22</li> <li>- LL_EXTI_LINE_29</li> <li>- LL_EXTI_LINE_30</li> <li>- LL_EXTI_LINE_31</li> </ul>                                                                                                                                                                                                                                                                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• The configurable wakeup lines are edge-triggered. No glitch must be generated on these lines. If a rising edge on a configurable interrupt line occurs during a write operation in the EXTI_RTSR register, the pending bit is not set. Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.</li> <li>• Please check each device line mapping for EXTI Line availability</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• RTSR RTx LL_EXTI_EnableRisingTrig_0_31</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                       |

### LL\_EXTI\_EnableRisingTrig\_32\_63

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_EXTI_EnableRisingTrig_32_63<br/>(uint32_t ExtiLine)</code>                                                                                                                                                                                                                                                                                                                              |
| Function description                              | Enable ExtiLine Rising Edge Trigger for Lines in range 32 to 63.                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_35</li> <li>- LL_EXTI_LINE_36</li> <li>- LL_EXTI_LINE_37</li> <li>- LL_EXTI_LINE_38</li> </ul> </li> </ul>                                                                                                                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• The configurable wakeup lines are edge-triggered. No glitch must be generated on these lines. If a rising edge on a configurable interrupt line occurs during a write operation in the EXTI_RTSR register, the pending bit is not set. Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• RTSR2 RTx LL_EXTI_EnableRisingTrig_32_63</li> </ul>                                                                                                                                                                                                                                                                                                                         |

### LL\_EXTI\_DisableRisingTrig\_0\_31

|                      |                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_EXTI_DisableRisingTrig_0_31<br/>(uint32_t ExtiLine)</code>                                                                                                                                                                   |
| Function description | Disable ExtiLine Rising Edge Trigger for Lines in range 0 to 31.                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_0</li> <li>- LL_EXTI_LINE_1</li> <li>- LL_EXTI_LINE_2</li> </ul> </li> </ul> |

- LL\_EXTI\_LINE\_3
- LL\_EXTI\_LINE\_4
- LL\_EXTI\_LINE\_5
- LL\_EXTI\_LINE\_6
- LL\_EXTI\_LINE\_7
- LL\_EXTI\_LINE\_8
- LL\_EXTI\_LINE\_9
- LL\_EXTI\_LINE\_10
- LL\_EXTI\_LINE\_11
- LL\_EXTI\_LINE\_12
- LL\_EXTI\_LINE\_13
- LL\_EXTI\_LINE\_14
- LL\_EXTI\_LINE\_15
- LL\_EXTI\_LINE\_16
- LL\_EXTI\_LINE\_18
- LL\_EXTI\_LINE\_19
- LL\_EXTI\_LINE\_20
- LL\_EXTI\_LINE\_21
- LL\_EXTI\_LINE\_22
- LL\_EXTI\_LINE\_29
- LL\_EXTI\_LINE\_30
- LL\_EXTI\_LINE\_31

Return values

- **None**

Notes

- The configurable wakeup lines are edge-triggered. No glitch must be generated on these lines. If a rising edge on a configurable interrupt line occurs during a write operation in the EXTI\_RTSR register, the pending bit is not set. Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.
- Please check each device line mapping for EXTI Line availability
- RTSR RTx LL\_EXTI\_DisableRisingTrig\_0\_31

Reference Manual to  
LL API cross  
reference:

### **LL\_EXTI\_DisableRisingTrig\_32\_63**

Function name

**STATIC\_INLINE void LL\_EXTI\_DisableRisingTrig\_32\_63  
(uint32\_t ExtiLine)**

Function description

Disable ExtiLine Rising Edge Trigger for Lines in range 32 to 63.

Parameters

- **ExtiLine:** This parameter can be a combination of the following values:
  - LL\_EXTI\_LINE\_35
  - LL\_EXTI\_LINE\_36
  - LL\_EXTI\_LINE\_37
  - LL\_EXTI\_LINE\_38

Return values

- **None**

Notes

- The configurable wakeup lines are edge-triggered. No glitch must be generated on these lines. If a rising edge on a configurable interrupt line occurs during a write operation in



the EXTI\_RTSR register, the pending bit is not set. Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.

Reference Manual to  
LL API cross  
reference:

- RTSR2 RTx LL\_EXTI\_DisableRisingTrig\_32\_63

### **LL\_EXTI\_IsEnabledRisingTrig\_0\_31**

Function name

`__STATIC_INLINE uint32_t  
LL_EXTI_IsEnabledRisingTrig_0_31 (uint32_t ExtiLine)`

Function description

Check if rising edge trigger is enabled for Lines in range 0 to 31.

Parameters

- ExtiLine:** This parameter can be a combination of the following values:
  - LL\_EXTI\_LINE\_0
  - LL\_EXTI\_LINE\_1
  - LL\_EXTI\_LINE\_2
  - LL\_EXTI\_LINE\_3
  - LL\_EXTI\_LINE\_4
  - LL\_EXTI\_LINE\_5
  - LL\_EXTI\_LINE\_6
  - LL\_EXTI\_LINE\_7
  - LL\_EXTI\_LINE\_8
  - LL\_EXTI\_LINE\_9
  - LL\_EXTI\_LINE\_10
  - LL\_EXTI\_LINE\_11
  - LL\_EXTI\_LINE\_12
  - LL\_EXTI\_LINE\_13
  - LL\_EXTI\_LINE\_14
  - LL\_EXTI\_LINE\_15
  - LL\_EXTI\_LINE\_16
  - LL\_EXTI\_LINE\_18
  - LL\_EXTI\_LINE\_19
  - LL\_EXTI\_LINE\_20
  - LL\_EXTI\_LINE\_21
  - LL\_EXTI\_LINE\_22
  - LL\_EXTI\_LINE\_29
  - LL\_EXTI\_LINE\_30
  - LL\_EXTI\_LINE\_31

Return values

- State:** of bit (1 or 0).

Notes

- Please check each device line mapping for EXTI Line availability

Reference Manual to  
LL API cross  
reference:

- RTSR RTx LL\_EXTI\_IsEnabledRisingTrig\_0\_31

**LL\_EXTI\_IsEnabledRisingTrig\_32\_63**

|                                                   |                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t<br/>LL_EXTI_IsEnabledRisingTrig_32_63 (uint32_t ExtiLine)</code>                                                                                                                                                                                                    |
| Function description                              | Check if rising edge trigger is enabled for Lines in range 32 to 63.                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_35</li> <li>- LL_EXTI_LINE_36</li> <li>- LL_EXTI_LINE_37</li> <li>- LL_EXTI_LINE_38</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | RTSR2 RTx LL_EXTI_IsEnabledRisingTrig_32_63                                                                                                                                                                                                                                                       |

**LL\_EXTI\_EnableFallingTrig\_0\_31**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_EXTI_EnableFallingTrig_0_31<br/>(uint32_t ExtiLine)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Enable ExtiLine Falling Edge Trigger for Lines in range 0 to 31.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_0</li> <li>- LL_EXTI_LINE_1</li> <li>- LL_EXTI_LINE_2</li> <li>- LL_EXTI_LINE_3</li> <li>- LL_EXTI_LINE_4</li> <li>- LL_EXTI_LINE_5</li> <li>- LL_EXTI_LINE_6</li> <li>- LL_EXTI_LINE_7</li> <li>- LL_EXTI_LINE_8</li> <li>- LL_EXTI_LINE_9</li> <li>- LL_EXTI_LINE_10</li> <li>- LL_EXTI_LINE_11</li> <li>- LL_EXTI_LINE_12</li> <li>- LL_EXTI_LINE_13</li> <li>- LL_EXTI_LINE_14</li> <li>- LL_EXTI_LINE_15</li> <li>- LL_EXTI_LINE_16</li> <li>- LL_EXTI_LINE_18</li> <li>- LL_EXTI_LINE_19</li> <li>- LL_EXTI_LINE_20</li> <li>- LL_EXTI_LINE_21</li> <li>- LL_EXTI_LINE_22</li> <li>- LL_EXTI_LINE_29</li> <li>- LL_EXTI_LINE_30</li> <li>- LL_EXTI_LINE_31</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>The configurable wakeup lines are edge-triggered. No glitch must be generated on these lines. If a falling edge on a configurable interrupt line occurs during a write operation in the EXTI_FTSR register, the pending bit is not set. Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.</li> <li>Please check each device line mapping for EXTI Line availability</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>FTSR FTx LL_EXTI_EnableFallingTrig_0_31</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                     |

### LL\_EXTI\_EnableFallingTrig\_32\_63

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_EXTI_EnableFallingTrig_32_63(<br/>  uint32_t ExtiLine)</code>                                                                                                                                                                                                                                                                                                                         |
| Function description                        | Enable ExtiLine Falling Edge Trigger for Lines in range 32 to 63.                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>ExtiLine:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_35</li> <li>- LL_EXTI_LINE_36</li> <li>- LL_EXTI_LINE_37</li> <li>- LL_EXTI_LINE_38</li> </ul> </li> </ul>                                                                                                                     |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                       |
| Notes                                       | <ul style="list-style-type: none"> <li>The configurable wakeup lines are edge-triggered. No glitch must be generated on these lines. If a Falling edge on a configurable interrupt line occurs during a write operation in the EXTI_FTSR register, the pending bit is not set. Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>FTSR2 FTx LL_EXTI_EnableFallingTrig_32_63</li> </ul>                                                                                                                                                                                                                                                                                                                         |

### LL\_EXTI\_DisableFallingTrig\_0\_31

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_EXTI_DisableFallingTrig_0_31(<br/>  uint32_t ExtiLine)</code>                                                                                                                                                                                                                                                                                                                                   |
| Function description | Disable ExtiLine Falling Edge Trigger for Lines in range 0 to 31.                                                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li><b>ExtiLine:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_0</li> <li>- LL_EXTI_LINE_1</li> <li>- LL_EXTI_LINE_2</li> <li>- LL_EXTI_LINE_3</li> <li>- LL_EXTI_LINE_4</li> <li>- LL_EXTI_LINE_5</li> <li>- LL_EXTI_LINE_6</li> <li>- LL_EXTI_LINE_7</li> <li>- LL_EXTI_LINE_8</li> </ul> </li> </ul> |

- LL\_EXTI\_LINE\_9
- LL\_EXTI\_LINE\_10
- LL\_EXTI\_LINE\_11
- LL\_EXTI\_LINE\_12
- LL\_EXTI\_LINE\_13
- LL\_EXTI\_LINE\_14
- LL\_EXTI\_LINE\_15
- LL\_EXTI\_LINE\_16
- LL\_EXTI\_LINE\_18
- LL\_EXTI\_LINE\_19
- LL\_EXTI\_LINE\_20
- LL\_EXTI\_LINE\_21
- LL\_EXTI\_LINE\_22
- LL\_EXTI\_LINE\_29
- LL\_EXTI\_LINE\_30
- LL\_EXTI\_LINE\_31

Return values

- **None**

Notes

- The configurable wakeup lines are edge-triggered. No glitch must be generated on these lines. If a Falling edge on a configurable interrupt line occurs during a write operation in the EXTI\_FTSR register, the pending bit is not set. Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.
- Please check each device line mapping for EXTI Line availability

Reference Manual to  
LL API cross  
reference:

- FTSR FTx LL\_EXTI\_DisableFallingTrig\_0\_31

### **LL\_EXTI\_DisableFallingTrig\_32\_63**

Function name

**\_STATIC\_INLINE void LL\_EXTI\_DisableFallingTrig\_32\_63  
(uint32\_t ExtiLine)**

Function description

Disable ExtiLine Falling Edge Trigger for Lines in range 32 to 63.

Parameters

- **ExtiLine:** This parameter can be a combination of the following values:
  - LL\_EXTI\_LINE\_35
  - LL\_EXTI\_LINE\_36
  - LL\_EXTI\_LINE\_37
  - LL\_EXTI\_LINE\_38

Return values

- **None**

Notes

- The configurable wakeup lines are edge-triggered. No glitch must be generated on these lines. If a Falling edge on a configurable interrupt line occurs during a write operation in the EXTI\_FTSR register, the pending bit is not set. Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.

Reference Manual to  
LL API cross  
reference:

- FTSR2 FTx LL\_EXTI\_DisableFallingTrig\_32\_63

### **LL\_EXTI\_IsEnabledFallingTrig\_0\_31**

Function name

`__STATIC_INLINE uint32_t`

`LL_EXTI_IsEnabledFallingTrig_0_31 (uint32_t ExtiLine)`

Function description

Check if falling edge trigger is enabled for Lines in range 0 to 31.

Parameters

- **ExtiLine:** This parameter can be a combination of the following values:

- `LL_EXTI_LINE_0`
- `LL_EXTI_LINE_1`
- `LL_EXTI_LINE_2`
- `LL_EXTI_LINE_3`
- `LL_EXTI_LINE_4`
- `LL_EXTI_LINE_5`
- `LL_EXTI_LINE_6`
- `LL_EXTI_LINE_7`
- `LL_EXTI_LINE_8`
- `LL_EXTI_LINE_9`
- `LL_EXTI_LINE_10`
- `LL_EXTI_LINE_11`
- `LL_EXTI_LINE_12`
- `LL_EXTI_LINE_13`
- `LL_EXTI_LINE_14`
- `LL_EXTI_LINE_15`
- `LL_EXTI_LINE_16`
- `LL_EXTI_LINE_18`
- `LL_EXTI_LINE_19`
- `LL_EXTI_LINE_20`
- `LL_EXTI_LINE_21`
- `LL_EXTI_LINE_22`
- `LL_EXTI_LINE_29`
- `LL_EXTI_LINE_30`
- `LL_EXTI_LINE_31`

Return values

- **State:** of bit (1 or 0).

Notes

- Please check each device line mapping for EXTI Line availability

Reference Manual to  
LL API cross  
reference:

- FTSR FTx LL\_EXTI\_IsEnabledFallingTrig\_0\_31

Function name

`__STATIC_INLINE uint32_t`

`LL_EXTI_IsEnabledFallingTrig_32_63 (uint32_t ExtiLine)`

Function description

Check if falling edge trigger is enabled for Lines in range 32 to 63.

Parameters

- **ExtiLine:** This parameter can be a combination of the following values:

- LL\_EXTI\_LINE\_35
  - LL\_EXTI\_LINE\_36
  - LL\_EXTI\_LINE\_37
  - LL\_EXTI\_LINE\_38
- Return values      • **State:** of bit (1 or 0).
- Reference Manual to  
LL API cross  
reference:      • FTSR2 FTx LL\_EXTI\_IsEnabledFallingTrig\_32\_63

### **LL\_EXTI\_GenerateSWI\_0\_31**

- Function name      **\_STATIC\_INLINE void LL\_EXTI\_GenerateSWI\_0\_31 (uint32\_t ExtiLine)**
- Function description      Generate a software Interrupt Event for Lines in range 0 to 31.
- Parameters      • **ExtiLine:** This parameter can be a combination of the following values:
  - LL\_EXTI\_LINE\_0
  - LL\_EXTI\_LINE\_1
  - LL\_EXTI\_LINE\_2
  - LL\_EXTI\_LINE\_3
  - LL\_EXTI\_LINE\_4
  - LL\_EXTI\_LINE\_5
  - LL\_EXTI\_LINE\_6
  - LL\_EXTI\_LINE\_7
  - LL\_EXTI\_LINE\_8
  - LL\_EXTI\_LINE\_9
  - LL\_EXTI\_LINE\_10
  - LL\_EXTI\_LINE\_11
  - LL\_EXTI\_LINE\_12
  - LL\_EXTI\_LINE\_13
  - LL\_EXTI\_LINE\_14
  - LL\_EXTI\_LINE\_15
  - LL\_EXTI\_LINE\_16
  - LL\_EXTI\_LINE\_18
  - LL\_EXTI\_LINE\_19
  - LL\_EXTI\_LINE\_20
  - LL\_EXTI\_LINE\_21
  - LL\_EXTI\_LINE\_22
  - LL\_EXTI\_LINE\_29
  - LL\_EXTI\_LINE\_30
  - LL\_EXTI\_LINE\_31
- Return values      • **None**
- Notes      • If the interrupt is enabled on this line in the EXTI\_IMR, writing a 1 to this bit when it is at '0' sets the corresponding pending bit in EXTI\_PR resulting in an interrupt request generation. This bit is cleared by clearing the corresponding bit in the EXTI\_PR register (by writing a 1 into the bit)
  - Please check each device line mapping for EXTI Line availability

Reference Manual to  
LL API cross  
reference:

- SWIER SWIx LL\_EXTI\_GenerateSWI\_0\_31

### **LL\_EXTI\_GenerateSWI\_32\_63**

Function name

**\_STATIC\_INLINE void LL\_EXTI\_GenerateSWI\_32\_63  
(uint32\_t ExtiLine)**

Function description

Generate a software Interrupt Event for Lines in range 32 to 63.

Parameters

- **ExtiLine:** This parameter can be a combination of the following values:
  - LL\_EXTI\_LINE\_35
  - LL\_EXTI\_LINE\_36
  - LL\_EXTI\_LINE\_37
  - LL\_EXTI\_LINE\_38

Return values

- **None**

Notes

- If the interrupt is enabled on this line inthe EXTI\_IMR2, writing a 1 to this bit when it is at '0' sets the corresponding pending bit in EXTI\_PR2 resulting in an interrupt request generation. This bit is cleared by clearing the corresponding bit in the EXTI\_PR2 register (by writing a 1 into the bit)

Reference Manual to  
LL API cross  
reference:

- SWIER2 SWIx LL\_EXTI\_GenerateSWI\_32\_63

### **LL\_EXTI\_IsActiveFlag\_0\_31**

Function name

**\_STATIC\_INLINE uint32\_t LL\_EXTI\_IsActiveFlag\_0\_31  
(uint32\_t ExtiLine)**

Function description

Check if the ExtLine Flag is set or not for Lines in range 0 to 31.

Parameters

- **ExtiLine:** This parameter can be a combination of the following values:
  - LL\_EXTI\_LINE\_0
  - LL\_EXTI\_LINE\_1
  - LL\_EXTI\_LINE\_2
  - LL\_EXTI\_LINE\_3
  - LL\_EXTI\_LINE\_4
  - LL\_EXTI\_LINE\_5
  - LL\_EXTI\_LINE\_6
  - LL\_EXTI\_LINE\_7
  - LL\_EXTI\_LINE\_8
  - LL\_EXTI\_LINE\_9
  - LL\_EXTI\_LINE\_10
  - LL\_EXTI\_LINE\_11
  - LL\_EXTI\_LINE\_12
  - LL\_EXTI\_LINE\_13
  - LL\_EXTI\_LINE\_14
  - LL\_EXTI\_LINE\_15
  - LL\_EXTI\_LINE\_16
  - LL\_EXTI\_LINE\_18

- LL\_EXTI\_LINE\_19
- LL\_EXTI\_LINE\_20
- LL\_EXTI\_LINE\_21
- LL\_EXTI\_LINE\_22
- LL\_EXTI\_LINE\_29
- LL\_EXTI\_LINE\_30
- LL\_EXTI\_LINE\_31

Return values

- **State:** of bit (1 or 0).

Notes

- This bit is set when the selected edge event arrives on the interrupt line. This bit is cleared by writing a 1 to the bit.
- Please check each device line mapping for EXTI Line availability

Reference Manual to  
LL API cross  
reference:

- PR PIFx LL\_EXTI\_IsActiveFlag\_0\_31

### **LL\_EXTI\_IsActiveFlag\_32\_63**

Function name

`_STATIC_INLINE uint32_t LL_EXTI_IsActiveFlag_32_63  
(uint32_t ExtiLine)`

Function description

Check if the ExtLine Flag is set or not for Lines in range 32 to 63.

Parameters

- **ExtiLine:** This parameter can be a combination of the following values:
  - LL\_EXTI\_LINE\_35
  - LL\_EXTI\_LINE\_36
  - LL\_EXTI\_LINE\_37
  - LL\_EXTI\_LINE\_38

Return values

- **State:** of bit (1 or 0).

Notes

- This bit is set when the selected edge event arrives on the interrupt line. This bit is cleared by writing a 1 to the bit.

Reference Manual to  
LL API cross  
reference:

- PR2 PIFx LL\_EXTI\_IsActiveFlag\_32\_63

### **LL\_EXTI\_ReadFlag\_0\_31**

Function name

`_STATIC_INLINE uint32_t LL_EXTI_ReadFlag_0_31 (uint32_t  
ExtiLine)`

Function description

Read ExtLine Combination Flag for Lines in range 0 to 31.

Parameters

- **ExtiLine:** This parameter can be a combination of the following values:
  - LL\_EXTI\_LINE\_0
  - LL\_EXTI\_LINE\_1
  - LL\_EXTI\_LINE\_2
  - LL\_EXTI\_LINE\_3
  - LL\_EXTI\_LINE\_4
  - LL\_EXTI\_LINE\_5
  - LL\_EXTI\_LINE\_6

- LL\_EXTI\_LINE\_7
- LL\_EXTI\_LINE\_8
- LL\_EXTI\_LINE\_9
- LL\_EXTI\_LINE\_10
- LL\_EXTI\_LINE\_11
- LL\_EXTI\_LINE\_12
- LL\_EXTI\_LINE\_13
- LL\_EXTI\_LINE\_14
- LL\_EXTI\_LINE\_15
- LL\_EXTI\_LINE\_16
- LL\_EXTI\_LINE\_18
- LL\_EXTI\_LINE\_19
- LL\_EXTI\_LINE\_20
- LL\_EXTI\_LINE\_21
- LL\_EXTI\_LINE\_22
- LL\_EXTI\_LINE\_29
- LL\_EXTI\_LINE\_30
- LL\_EXTI\_LINE\_31

|                                                   |                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>@note:</b> This bit is set when the selected edge event arrives on the interrupt</li> </ul>                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit is set when the selected edge event arrives on the interrupt line. This bit is cleared by writing a 1 to the bit.</li> <li>• Please check each device line mapping for EXTI Line availability</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• PR PIFx LL_EXTI_ReadFlag_0_31</li> </ul>                                                                                                                                                                          |

### LL\_EXTI\_ReadFlag\_32\_63

|                                                   |                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_EXTI_ReadFlag_32_63(<br/>  uint32_t ExtiLine)</code>                                                                                                                                                                                                            |
| Function description                              | Read ExtLine Combination Flag for Lines in range 32 to 63.                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_35</li> <li>- LL_EXTI_LINE_36</li> <li>- LL_EXTI_LINE_37</li> <li>- LL_EXTI_LINE_38</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>@note:</b> This bit is set when the selected edge event arrives on the interrupt</li> </ul>                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit is set when the selected edge event arrives on the interrupt line. This bit is cleared by writing a 1 to the bit.</li> </ul>                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• PR2 PIFx LL_EXTI_ReadFlag_32_63</li> </ul>                                                                                                                                                                                                               |

**LL\_EXTI\_ClearFlag\_0\_31**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_EXTI_ClearFlag_0_31 (uint32_t ExtiLine)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description                              | Clear ExtLine Flags for Lines in range 0 to 31.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- <code>LL_EXTI_LINE_0</code></li> <li>- <code>LL_EXTI_LINE_1</code></li> <li>- <code>LL_EXTI_LINE_2</code></li> <li>- <code>LL_EXTI_LINE_3</code></li> <li>- <code>LL_EXTI_LINE_4</code></li> <li>- <code>LL_EXTI_LINE_5</code></li> <li>- <code>LL_EXTI_LINE_6</code></li> <li>- <code>LL_EXTI_LINE_7</code></li> <li>- <code>LL_EXTI_LINE_8</code></li> <li>- <code>LL_EXTI_LINE_9</code></li> <li>- <code>LL_EXTI_LINE_10</code></li> <li>- <code>LL_EXTI_LINE_11</code></li> <li>- <code>LL_EXTI_LINE_12</code></li> <li>- <code>LL_EXTI_LINE_13</code></li> <li>- <code>LL_EXTI_LINE_14</code></li> <li>- <code>LL_EXTI_LINE_15</code></li> <li>- <code>LL_EXTI_LINE_16</code></li> <li>- <code>LL_EXTI_LINE_18</code></li> <li>- <code>LL_EXTI_LINE_19</code></li> <li>- <code>LL_EXTI_LINE_20</code></li> <li>- <code>LL_EXTI_LINE_21</code></li> <li>- <code>LL_EXTI_LINE_22</code></li> <li>- <code>LL_EXTI_LINE_29</code></li> <li>- <code>LL_EXTI_LINE_30</code></li> <li>- <code>LL_EXTI_LINE_31</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit is set when the selected edge event arrives on the interrupt line. This bit is cleared by writing a 1 to the bit.</li> <li>• Please check each device line mapping for EXTI Line availability</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• PR PIFx LL_EXTI_ClearFlag_0_31</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

**LL\_EXTI\_ClearFlag\_32\_63**

|                      |                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_EXTI_ClearFlag_32_63 (uint32_t ExtiLine)</code>                                                                                                                                                                               |
| Function description | Clear ExtLine Flags for Lines in range 32 to 63.                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ExtiLine:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- <code>LL_EXTI_LINE_35</code></li> <li>- <code>LL_EXTI_LINE_36</code></li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_EXTI_LINE_37</li> <li>- LL_EXTI_LINE_38</li> </ul>                                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit is set when the selected edge event arrives on the interrupt line. This bit is cleared by writing a 1 to the bit.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• PR2 PIFx LL_EXTI_ClearFlag_32_63</li> </ul>                                                                                           |

### LL\_EXTI\_Init

|                      |                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t LL_EXTI_Init (LL_EXTI_InitTypeDef * EXTI_InitStruct)</b>                                                                                                                                                        |
| Function description | Initialize the EXTI registers according to the specified parameters in EXTI_InitStruct.                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>EXTI_InitStruct:</b> pointer to a LL_EXTI_InitTypeDef structure.</li> </ul>                                                                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>- SUCCESS: EXTI registers are initialized</li> <li>- ERROR: not applicable</li> </ul> </li> </ul> |

### LL\_EXTI\_DeInit

|                      |                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>uint32_t LL_EXTI_DeInit (void )</b>                                                                                                                                                                                         |
| Function description | De-initialize the EXTI registers to their default reset values.                                                                                                                                                                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>- SUCCESS: EXTI registers are de-initialized</li> <li>- ERROR: not applicable</li> </ul> </li> </ul> |

### LL\_EXTI\_StructInit

|                      |                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_EXTI_StructInit (LL_EXTI_InitTypeDef *<br/>EXTI_InitStruct)</b>                                              |
| Function description | Set each LL_EXTI_InitTypeDef field to default value.                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>EXTI_InitStruct:</b> Pointer to a LL_EXTI_InitTypeDef structure.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                         |

## 66.3 EXTI Firmware driver defines

### 66.3.1 EXTI

#### *LINE*

|                |                 |
|----------------|-----------------|
| LL_EXTI_LINE_0 | Extended line 0 |
| LL_EXTI_LINE_1 | Extended line 1 |
| LL_EXTI_LINE_2 | Extended line 2 |
| LL_EXTI_LINE_3 | Extended line 3 |
| LL_EXTI_LINE_4 | Extended line 4 |

|                        |                                |
|------------------------|--------------------------------|
| LL_EXTI_LINE_5         | Extended line 5                |
| LL_EXTI_LINE_6         | Extended line 6                |
| LL_EXTI_LINE_7         | Extended line 7                |
| LL_EXTI_LINE_8         | Extended line 8                |
| LL_EXTI_LINE_9         | Extended line 9                |
| LL_EXTI_LINE_10        | Extended line 10               |
| LL_EXTI_LINE_11        | Extended line 11               |
| LL_EXTI_LINE_12        | Extended line 12               |
| LL_EXTI_LINE_13        | Extended line 13               |
| LL_EXTI_LINE_14        | Extended line 14               |
| LL_EXTI_LINE_15        | Extended line 15               |
| LL_EXTI_LINE_16        | Extended line 16               |
| LL_EXTI_LINE_17        | Extended line 17               |
| LL_EXTI_LINE_18        | Extended line 18               |
| LL_EXTI_LINE_19        | Extended line 19               |
| LL_EXTI_LINE_20        | Extended line 20               |
| LL_EXTI_LINE_21        | Extended line 21               |
| LL_EXTI_LINE_22        | Extended line 22               |
| LL_EXTI_LINE_23        | Extended line 23               |
| LL_EXTI_LINE_24        | Extended line 24               |
| LL_EXTI_LINE_25        | Extended line 25               |
| LL_EXTI_LINE_26        | Extended line 26               |
| LL_EXTI_LINE_28        | Extended line 28               |
| LL_EXTI_LINE_29        | Extended line 29               |
| LL_EXTI_LINE_30        | Extended line 30               |
| LL_EXTI_LINE_31        | Extended line 31               |
| LL_EXTI_LINE_ALL_0_31  | All Extended line not reserved |
| LL_EXTI_LINE_32        | Extended line 32               |
| LL_EXTI_LINE_33        | Extended line 33               |
| LL_EXTI_LINE_34        | Extended line 34               |
| LL_EXTI_LINE_35        | Extended line 35               |
| LL_EXTI_LINE_ALL_32_63 | All Extended line not reserved |
| LL_EXTI_LINE_ALL       | All Extended line              |
| LL_EXTI_LINE_NONE      | None Extended line             |
| <b>Mode</b>            |                                |
| LL_EXTI_MODE_IT        | Interrupt Mode                 |

---

|                       |                        |
|-----------------------|------------------------|
| LL_EXTI_MODE_EVENT    | Event Mode             |
| LL_EXTI_MODE_IT_EVENT | Interrupt & Event Mode |

**Edge Trigger**

|                                |                               |
|--------------------------------|-------------------------------|
| LL_EXTI_TRIGGER_NONE           | No Trigger Mode               |
| LL_EXTI_TRIGGER_RISING         | Trigger Rising Mode           |
| LL_EXTI_TRIGGER_FALLING        | Trigger Falling Mode          |
| LL_EXTI_TRIGGER_RISING_FALLING | Trigger Rising & Falling Mode |

**Common Write and read registers Macros**

|                  |                                                                                 |
|------------------|---------------------------------------------------------------------------------|
| LL_EXTI_WriteReg | <b>Description:</b>                                                             |
|                  | <ul style="list-style-type: none"><li>Write a value in EXTI register.</li></ul> |

**Parameters:**

- `__REG__`: Register to be written
- `__VALUE__`: Value to be written in the register

**Return value:**

- None

|                 |                                                                                |
|-----------------|--------------------------------------------------------------------------------|
| LL_EXTI_ReadReg | <b>Description:</b>                                                            |
|                 | <ul style="list-style-type: none"><li>Read a value in EXTI register.</li></ul> |

**Parameters:**

- `__REG__`: Register to be read

**Return value:**

- Register: value

## 67 LL GPIO Generic Driver

### 67.1 GPIO Firmware driver registers structures

#### 67.1.1 LL\_GPIO\_InitTypeDef

##### Data Fields

- *uint32\_t Pin*
- *uint32\_t Mode*
- *uint32\_t Speed*
- *uint32\_t OutputType*
- *uint32\_t Pull*
- *uint32\_t Alternate*

##### Field Documentation

- ***uint32\_t LL\_GPIO\_InitTypeDef::Pin***  
Specifies the GPIO pins to be configured. This parameter can be any value of [\*\*GPIO\\_LL\\_EC\\_PIN\*\*](#)
- ***uint32\_t LL\_GPIO\_InitTypeDef::Mode***  
Specifies the operating mode for the selected pins. This parameter can be a value of [\*\*GPIO\\_LL\\_EC\\_MODE\*\*](#).GPIO HW configuration can be modified afterwards using unitary function [\*\*LL\\_GPIO\\_SetPinMode\(\)\*\*](#).
- ***uint32\_t LL\_GPIO\_InitTypeDef::Speed***  
Specifies the speed for the selected pins. This parameter can be a value of [\*\*GPIO\\_LL\\_EC\\_SPEED\*\*](#).GPIO HW configuration can be modified afterwards using unitary function [\*\*LL\\_GPIO\\_SetPinSpeed\(\)\*\*](#).
- ***uint32\_t LL\_GPIO\_InitTypeDef::OutputType***  
Specifies the operating output type for the selected pins. This parameter can be a value of [\*\*GPIO\\_LL\\_EC\\_OUTPUT\*\*](#).GPIO HW configuration can be modified afterwards using unitary function [\*\*LL\\_GPIO\\_SetPinOutputType\(\)\*\*](#).
- ***uint32\_t LL\_GPIO\_InitTypeDef::Pull***  
Specifies the operating Pull-up/Pull down for the selected pins. This parameter can be a value of [\*\*GPIO\\_LL\\_EC\\_PULL\*\*](#).GPIO HW configuration can be modified afterwards using unitary function [\*\*LL\\_GPIO\\_SetPinPull\(\)\*\*](#).
- ***uint32\_t LL\_GPIO\_InitTypeDef::Alternate***  
Specifies the Peripheral to be connected to the selected pins. This parameter can be a value of [\*\*GPIO\\_LL\\_EC\\_AF\*\*](#).GPIO HW configuration can be modified afterwards using unitary function [\*\*LL\\_GPIO\\_SetAFPin\\_0\\_7\(\)\*\*](#) and [\*\*LL\\_GPIO\\_SetAFPin\\_8\\_15\(\)\*\*](#).

### 67.2 GPIO Firmware driver API description

#### 67.2.1 Detailed description of functions

##### LL\_GPIO\_SetPinMode

Function name      `_STATIC_INLINE void LL_GPIO_SetPinMode (GPIO_TypeDef * GPIOx, uint32_t Pin, uint32_t Mode)`

Function description      Configure gpio mode for a dedicated pin on dedicated port.

Parameters     
 

- **GPIOx:** GPIO Port
- **Pin:** This parameter can be one of the following values:

- LL\_GPIO\_PIN\_0
- LL\_GPIO\_PIN\_1
- LL\_GPIO\_PIN\_2
- LL\_GPIO\_PIN\_3
- LL\_GPIO\_PIN\_4
- LL\_GPIO\_PIN\_5
- LL\_GPIO\_PIN\_6
- LL\_GPIO\_PIN\_7
- LL\_GPIO\_PIN\_8
- LL\_GPIO\_PIN\_9
- LL\_GPIO\_PIN\_10
- LL\_GPIO\_PIN\_11
- LL\_GPIO\_PIN\_12
- LL\_GPIO\_PIN\_13
- LL\_GPIO\_PIN\_14
- LL\_GPIO\_PIN\_15
- **Mode:** This parameter can be one of the following values:
  - LL\_GPIO\_MODE\_INPUT
  - LL\_GPIO\_MODE\_OUTPUT
  - LL\_GPIO\_MODE\_ALTERNATE
  - LL\_GPIO\_MODE\_ANALOG

Return values

- **None**

Notes

- I/O mode can be Input mode, General purpose output, Alternate function mode or Analog.
- Warning: only one pin can be passed as parameter.

Reference Manual to  
LL API cross  
reference:

- MODER MODEy LL\_GPIO\_SetPinMode

### **LL\_GPIO\_GetPinMode**

Function name

`__STATIC_INLINE uint32_t LL_GPIO_GetPinMode  
(GPIO_TypeDef * GPIOx, uint32_t Pin)`

Function description

Return gpio mode for a dedicated pin on dedicated port.

Parameters

- **GPIOx:** GPIO Port
- **Pin:** This parameter can be one of the following values:
  - LL\_GPIO\_PIN\_0
  - LL\_GPIO\_PIN\_1
  - LL\_GPIO\_PIN\_2
  - LL\_GPIO\_PIN\_3
  - LL\_GPIO\_PIN\_4
  - LL\_GPIO\_PIN\_5
  - LL\_GPIO\_PIN\_6
  - LL\_GPIO\_PIN\_7
  - LL\_GPIO\_PIN\_8
  - LL\_GPIO\_PIN\_9
  - LL\_GPIO\_PIN\_10
  - LL\_GPIO\_PIN\_11
  - LL\_GPIO\_PIN\_12
  - LL\_GPIO\_PIN\_13

|                                                   |                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_GPIO_PIN_14</li> <li>- LL_GPIO_PIN_15</li> </ul>                                                                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_GPIO_MODE_INPUT</li> <li>- LL_GPIO_MODE_OUTPUT</li> <li>- LL_GPIO_MODE_ALTERNATE</li> <li>- LL_GPIO_MODE_ANALOG</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• I/O mode can be Input mode, General purpose output, Alternate function mode or Analog.</li> <li>• Warning: only one pin can be passed as parameter.</li> </ul>                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MODER MODEy LL_GPIO_SetPinMode</li> </ul>                                                                                                                                                                                                               |

### LL\_GPIO\_SetPinOutputType

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <pre><code>__STATIC_INLINE void LL_GPIO_SetPinOutputType (GPIO_TypeDef * GPIOx, uint32_t PinMask, uint32_t OutputType)</code></pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Configure gpio output type for several pins on dedicated port.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>PinMask:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>- LL_GPIO_PIN_0</li> <li>- LL_GPIO_PIN_1</li> <li>- LL_GPIO_PIN_2</li> <li>- LL_GPIO_PIN_3</li> <li>- LL_GPIO_PIN_4</li> <li>- LL_GPIO_PIN_5</li> <li>- LL_GPIO_PIN_6</li> <li>- LL_GPIO_PIN_7</li> <li>- LL_GPIO_PIN_8</li> <li>- LL_GPIO_PIN_9</li> <li>- LL_GPIO_PIN_10</li> <li>- LL_GPIO_PIN_11</li> <li>- LL_GPIO_PIN_12</li> <li>- LL_GPIO_PIN_13</li> <li>- LL_GPIO_PIN_14</li> <li>- LL_GPIO_PIN_15</li> <li>- LL_GPIO_PIN_ALL</li> </ul> </li> <li>• <b>OutputType:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_GPIO_OUTPUT_PUSHPULL</li> <li>- LL_GPIO_OUTPUT_OPENDRAIN</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• Output type as to be set when gpio pin is in output or alternate modes. Possible type are Push-pull or Open-drain.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

Reference Manual to  
LL API cross  
reference:

- OTYPER OTy LL\_GPIO\_SetPinOutputType

### **LL\_GPIO\_GetPinOutputType**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_GPIO_GetPinOutputType<br/>(GPIO_TypeDef * GPIOx, uint32_t Pin)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Return gpio output type for several pins on dedicated port.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>Pin:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_GPIO_PIN_0</li> <li>- LL_GPIO_PIN_1</li> <li>- LL_GPIO_PIN_2</li> <li>- LL_GPIO_PIN_3</li> <li>- LL_GPIO_PIN_4</li> <li>- LL_GPIO_PIN_5</li> <li>- LL_GPIO_PIN_6</li> <li>- LL_GPIO_PIN_7</li> <li>- LL_GPIO_PIN_8</li> <li>- LL_GPIO_PIN_9</li> <li>- LL_GPIO_PIN_10</li> <li>- LL_GPIO_PIN_11</li> <li>- LL_GPIO_PIN_12</li> <li>- LL_GPIO_PIN_13</li> <li>- LL_GPIO_PIN_14</li> <li>- LL_GPIO_PIN_15</li> <li>- LL_GPIO_PIN_ALL</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_GPIO_OUTPUT_PUSHPULL</li> <li>- LL_GPIO_OUTPUT_OPENDRAIN</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• Output type as to be set when gpio pin is in output or alternate modes. Possible type are Push-pull or Open-drain.</li> <li>• Warning: only one pin can be passed as parameter.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OTYPER OTy LL_GPIO_SetPinOutputType</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

### **LL\_GPIO\_SetPinSpeed**

|                      |                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_GPIO_SetPinSpeed<br/>(GPIO_TypeDef * GPIOx, uint32_t Pin, uint32_t Speed)</code>                                                                                                                                                                                                                |
| Function description | Configure gpio speed for a dedicated pin on dedicated port.                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>Pin:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_GPIO_PIN_0</li> <li>- LL_GPIO_PIN_1</li> <li>- LL_GPIO_PIN_2</li> <li>- LL_GPIO_PIN_3</li> <li>- LL_GPIO_PIN_4</li> </ul> </li> </ul> |

- LL\_GPIO\_PIN\_5
  - LL\_GPIO\_PIN\_6
  - LL\_GPIO\_PIN\_7
  - LL\_GPIO\_PIN\_8
  - LL\_GPIO\_PIN\_9
  - LL\_GPIO\_PIN\_10
  - LL\_GPIO\_PIN\_11
  - LL\_GPIO\_PIN\_12
  - LL\_GPIO\_PIN\_13
  - LL\_GPIO\_PIN\_14
  - LL\_GPIO\_PIN\_15
  - **Speed:** This parameter can be one of the following values:
    - LL\_GPIO\_SPEED\_FREQ\_LOW
    - LL\_GPIO\_SPEED\_FREQ\_MEDIUM
    - LL\_GPIO\_SPEED\_FREQ\_HIGH
- |                                                   |                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | • <b>None</b>                                                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• I/O speed can be Low, Medium, Fast or High speed.</li> <li>• Warning: only one pin can be passed as parameter.</li> <li>• Refer to datasheet for frequency specifications and the power supply and load conditions for each speed.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OSPEEDR OSPEEDy LL_GPIO_SetPinSpeed</li> </ul>                                                                                                                                                                                                |

## LL\_GPIO\_SetPinSpeed

- |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>STATIC_INLINE uint32_t LL_GPIO_SetPinSpeed<br/>(GPIO_TypeDef * GPIOx, uint32_t Pin)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Return gpio speed for a dedicated pin on dedicated port.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>Pin:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_GPIO_PIN_0</li> <li>- LL_GPIO_PIN_1</li> <li>- LL_GPIO_PIN_2</li> <li>- LL_GPIO_PIN_3</li> <li>- LL_GPIO_PIN_4</li> <li>- LL_GPIO_PIN_5</li> <li>- LL_GPIO_PIN_6</li> <li>- LL_GPIO_PIN_7</li> <li>- LL_GPIO_PIN_8</li> <li>- LL_GPIO_PIN_9</li> <li>- LL_GPIO_PIN_10</li> <li>- LL_GPIO_PIN_11</li> <li>- LL_GPIO_PIN_12</li> <li>- LL_GPIO_PIN_13</li> <li>- LL_GPIO_PIN_14</li> <li>- LL_GPIO_PIN_15</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_GPIO_SPEED_FREQ_LOW</li> <li>- LL_GPIO_SPEED_FREQ_MEDIUM</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                       |

|                                                   |                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | – LL_GPIO_SPEED_FREQ_HIGH                                                                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• I/O speed can be Low, Medium, Fast or High speed.</li> <li>• Warning: only one pin can be passed as parameter.</li> <li>• Refer to datasheet for frequency specifications and the power supply and load conditions for each speed.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OSPEEDR OSPEEDY LL_GPIO_GetPinSpeed</li> </ul>                                                                                                                                                                                                |

### LL\_GPIO\_SetPinPull

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_GPIO_SetPinPull (GPIO_TypeDef * GPIOx, uint32_t Pin, uint32_t Pull)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Configure gpio pull-up or pull-down for a dedicated pin on a dedicated port.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>Pin:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_GPIO_PIN_0</li> <li>– LL_GPIO_PIN_1</li> <li>– LL_GPIO_PIN_2</li> <li>– LL_GPIO_PIN_3</li> <li>– LL_GPIO_PIN_4</li> <li>– LL_GPIO_PIN_5</li> <li>– LL_GPIO_PIN_6</li> <li>– LL_GPIO_PIN_7</li> <li>– LL_GPIO_PIN_8</li> <li>– LL_GPIO_PIN_9</li> <li>– LL_GPIO_PIN_10</li> <li>– LL_GPIO_PIN_11</li> <li>– LL_GPIO_PIN_12</li> <li>– LL_GPIO_PIN_13</li> <li>– LL_GPIO_PIN_14</li> <li>– LL_GPIO_PIN_15</li> </ul> </li> <li>• <b>Pull:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_GPIO_PULL_NO</li> <li>– LL_GPIO_PULL_UP</li> <li>– LL_GPIO_PULL_DOWN</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• Warning: only one pin can be passed as parameter.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• PUPDR PUPDy LL_GPIO_SetPinPull</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

### LL\_GPIO\_SetPinPull

|                      |                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_GPIO_SetPinPull (GPIO_TypeDef * GPIOx, uint32_t Pin)</code></b> |
| Function description | Return gpio pull-up or pull-down for a dedicated pin on a dedicated port.                            |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>GPIOx:</b> GPIO Port</li> <li><b>Pin:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_GPIO_PIN_0</li> <li>- LL_GPIO_PIN_1</li> <li>- LL_GPIO_PIN_2</li> <li>- LL_GPIO_PIN_3</li> <li>- LL_GPIO_PIN_4</li> <li>- LL_GPIO_PIN_5</li> <li>- LL_GPIO_PIN_6</li> <li>- LL_GPIO_PIN_7</li> <li>- LL_GPIO_PIN_8</li> <li>- LL_GPIO_PIN_9</li> <li>- LL_GPIO_PIN_10</li> <li>- LL_GPIO_PIN_11</li> <li>- LL_GPIO_PIN_12</li> <li>- LL_GPIO_PIN_13</li> <li>- LL_GPIO_PIN_14</li> <li>- LL_GPIO_PIN_15</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_GPIO_PULL_NO</li> <li>- LL_GPIO_PULL_UP</li> <li>- LL_GPIO_PULL_DOWN</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>Warning: only one pin can be passed as parameter.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>PUPDR PUPDy LL_GPIO_GetPinPull</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

## LL\_GPIO\_SetAFPin\_0\_7

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_GPIO_SetAFPin_0_7<br/>(GPIO_TypeDef * GPIOx, uint32_t Pin, uint32_t Alternate)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Configure gpio alternate function of a dedicated pin from 0 to 7 for a dedicated port.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>GPIOx:</b> GPIO Port</li> <li><b>Pin:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_GPIO_PIN_0</li> <li>- LL_GPIO_PIN_1</li> <li>- LL_GPIO_PIN_2</li> <li>- LL_GPIO_PIN_3</li> <li>- LL_GPIO_PIN_4</li> <li>- LL_GPIO_PIN_5</li> <li>- LL_GPIO_PIN_6</li> <li>- LL_GPIO_PIN_7</li> </ul> </li> <li><b>Alternate:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_GPIO_AF_0</li> <li>- LL_GPIO_AF_1</li> <li>- LL_GPIO_AF_2</li> <li>- LL_GPIO_AF_3</li> <li>- LL_GPIO_AF_4</li> </ul> </li> </ul> |

- LL\_GPIO\_AF\_5
- LL\_GPIO\_AF\_6
- LL\_GPIO\_AF\_7
- LL\_GPIO\_AF\_8
- LL\_GPIO\_AF\_9
- LL\_GPIO\_AF\_10
- LL\_GPIO\_AF\_11
- LL\_GPIO\_AF\_12
- LL\_GPIO\_AF\_13
- LL\_GPIO\_AF\_14
- LL\_GPIO\_AF\_15

**Return values**

- **None**

**Notes**

- Possible values are from AF0 to AF15 depending on target.
- Warning: only one pin can be passed as parameter.

**Reference Manual to  
LL API cross  
reference:**

- AFRL AFSELy LL\_GPIO\_SetAFPin\_0\_7

**LL\_GPIO\_GetAFPin\_0\_7****Function name**

**\_STATIC\_INLINE uint32\_t LL\_GPIO\_GetAFPin\_0\_7  
(GPIO\_TypeDef \* GPIOx, uint32\_t Pin)**

**Function description**

Return gpio alternate function of a dedicated pin from 0 to 7 for a dedicated port.

**Parameters**

- **GPIOx:** GPIO Port
- **Pin:** This parameter can be one of the following values:
  - LL\_GPIO\_PIN\_0
  - LL\_GPIO\_PIN\_1
  - LL\_GPIO\_PIN\_2
  - LL\_GPIO\_PIN\_3
  - LL\_GPIO\_PIN\_4
  - LL\_GPIO\_PIN\_5
  - LL\_GPIO\_PIN\_6
  - LL\_GPIO\_PIN\_7

**Return values**

- **Returned:** value can be one of the following values:
  - LL\_GPIO\_AF\_0
  - LL\_GPIO\_AF\_1
  - LL\_GPIO\_AF\_2
  - LL\_GPIO\_AF\_3
  - LL\_GPIO\_AF\_4
  - LL\_GPIO\_AF\_5
  - LL\_GPIO\_AF\_6
  - LL\_GPIO\_AF\_7
  - LL\_GPIO\_AF\_8
  - LL\_GPIO\_AF\_9
  - LL\_GPIO\_AF\_10
  - LL\_GPIO\_AF\_11
  - LL\_GPIO\_AF\_12
  - LL\_GPIO\_AF\_13
  - LL\_GPIO\_AF\_14

- LL\_GPIO\_AF\_15
  - AFRL AFSELy LL\_GPIO\_SetAFPin\_0\_7
- Reference Manual to  
LL API cross  
reference:

### **LL\_GPIO\_SetAFPin\_8\_15**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_GPIO_SetAFPin_8_15(<br/>GPIO_TypeDef * GPIOx, uint32_t Pin, uint32_t Alternate)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description                              | Configure gpio alternate function of a dedicated pin from 8 to 15 for a dedicated port.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>Pin:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_GPIO_PIN_8</li> <li>- LL_GPIO_PIN_9</li> <li>- LL_GPIO_PIN_10</li> <li>- LL_GPIO_PIN_11</li> <li>- LL_GPIO_PIN_12</li> <li>- LL_GPIO_PIN_13</li> <li>- LL_GPIO_PIN_14</li> <li>- LL_GPIO_PIN_15</li> </ul> </li> <li>• <b>Alternate:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_GPIO_AF_0</li> <li>- LL_GPIO_AF_1</li> <li>- LL_GPIO_AF_2</li> <li>- LL_GPIO_AF_3</li> <li>- LL_GPIO_AF_4</li> <li>- LL_GPIO_AF_5</li> <li>- LL_GPIO_AF_6</li> <li>- LL_GPIO_AF_7</li> <li>- LL_GPIO_AF_8</li> <li>- LL_GPIO_AF_9</li> <li>- LL_GPIO_AF_10</li> <li>- LL_GPIO_AF_11</li> <li>- LL_GPIO_AF_12</li> <li>- LL_GPIO_AF_13</li> <li>- LL_GPIO_AF_14</li> <li>- LL_GPIO_AF_15</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• Possible values are from AF0 to AF15 depending on target.</li> <li>• Warning: only one pin can be passed as parameter.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• AFRH AFSELy LL_GPIO_SetAFPin_8_15</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

**LL\_GPIO\_GetAFPin\_8\_15**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_GPIO_GetAFPin_8_15<br/>(GPIO_TypeDef * GPIOx, uint32_t Pin)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Return gpio alternate function of a dedicated pin from 8 to 15 for a dedicated port.                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>Pin:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_GPIO_PIN_8</li> <li>– LL_GPIO_PIN_9</li> <li>– LL_GPIO_PIN_10</li> <li>– LL_GPIO_PIN_11</li> <li>– LL_GPIO_PIN_12</li> <li>– LL_GPIO_PIN_13</li> <li>– LL_GPIO_PIN_14</li> <li>– LL_GPIO_PIN_15</li> </ul> </li> </ul>                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_GPIO_AF_0</li> <li>– LL_GPIO_AF_1</li> <li>– LL_GPIO_AF_2</li> <li>– LL_GPIO_AF_3</li> <li>– LL_GPIO_AF_4</li> <li>– LL_GPIO_AF_5</li> <li>– LL_GPIO_AF_6</li> <li>– LL_GPIO_AF_7</li> <li>– LL_GPIO_AF_8</li> <li>– LL_GPIO_AF_9</li> <li>– LL_GPIO_AF_10</li> <li>– LL_GPIO_AF_11</li> <li>– LL_GPIO_AF_12</li> <li>– LL_GPIO_AF_13</li> <li>– LL_GPIO_AF_14</li> <li>– LL_GPIO_AF_15</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• Possible values are from AF0 to AF15 depending on target.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• AFRH AFSELy LL_GPIO_GetAFPin_8_15</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

**LL\_GPIO\_LockPin**

|                      |                                                                                                                                                                                                                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_GPIO_LockPin (GPIO_TypeDef *<br/>GPIOx, uint32_t PinMask)</code>                                                                                                                                                                                                                     |
| Function description | Lock configuration of several pins for a dedicated port.                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>PinMask:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_GPIO_PIN_0</li> <li>– LL_GPIO_PIN_1</li> <li>– LL_GPIO_PIN_2</li> <li>– LL_GPIO_PIN_3</li> </ul> </li> </ul> |

- LL\_GPIO\_PIN\_4
- LL\_GPIO\_PIN\_5
- LL\_GPIO\_PIN\_6
- LL\_GPIO\_PIN\_7
- LL\_GPIO\_PIN\_8
- LL\_GPIO\_PIN\_9
- LL\_GPIO\_PIN\_10
- LL\_GPIO\_PIN\_11
- LL\_GPIO\_PIN\_12
- LL\_GPIO\_PIN\_13
- LL\_GPIO\_PIN\_14
- LL\_GPIO\_PIN\_15
- LL\_GPIO\_PIN\_ALL

Return values

- **None**

Notes

- When the lock sequence has been applied on a port bit, the value of this port bit can no longer be modified until the next reset.
- Each lock bit freezes a specific configuration register (control and alternate function registers).

Reference Manual to  
LL API cross  
reference:

- LCKR LCKK LL\_GPIO\_LockPin

### **LL\_GPIO\_IsPinLocked**

Function name

`_STATIC_INLINE uint32_t LL_GPIO_IsPinLocked  
(GPIO_TypeDef * GPIOx, uint32_t PinMask)`

Function description

Return 1 if all pins passed as parameter, of a dedicated port, are locked.

Parameters

- **GPIOx:** GPIO Port
- **PinMask:** This parameter can be a combination of the following values:
  - LL\_GPIO\_PIN\_0
  - LL\_GPIO\_PIN\_1
  - LL\_GPIO\_PIN\_2
  - LL\_GPIO\_PIN\_3
  - LL\_GPIO\_PIN\_4
  - LL\_GPIO\_PIN\_5
  - LL\_GPIO\_PIN\_6
  - LL\_GPIO\_PIN\_7
  - LL\_GPIO\_PIN\_8
  - LL\_GPIO\_PIN\_9
  - LL\_GPIO\_PIN\_10
  - LL\_GPIO\_PIN\_11
  - LL\_GPIO\_PIN\_12
  - LL\_GPIO\_PIN\_13
  - LL\_GPIO\_PIN\_14
  - LL\_GPIO\_PIN\_15
  - LL\_GPIO\_PIN\_ALL

---

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>LCKR LCKY LL_GPIO_IsPinLocked</li> </ul>  |

### LL\_GPIO\_IsAnyPinLocked

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_GPIO_IsAnyPinLocked<br/>(GPIO_TypeDef * GPIOx)</code> |
| Function description                              | Return 1 if one of the pin of a dedicated port is locked.                               |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>GPIOx:</b> GPIO Port</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>LCKR LCKK LL_GPIO_IsAnyPinLocked</li> </ul>      |

### LL\_GPIO\_ReadInputPort

|                                                   |                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_GPIO_ReadInputPort<br/>(GPIO_TypeDef * GPIOx)</code>      |
| Function description                              | Return full input data register value for a dedicated port.                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>GPIOx:</b> GPIO Port</li> </ul>                   |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Input:</b> data register value of port</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>IDR IDy LL_GPIO_ReadInputPort</li> </ul>             |

### LL\_GPIO\_IsInputPinSet

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_GPIO_IsInputPinSet<br/>(GPIO_TypeDef * GPIOx, uint32_t PinMask)</code>                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description | Return if input data level for several pins of dedicated port is high or low.                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li><b>GPIOx:</b> GPIO Port</li> <li><b>PinMask:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_GPIO_PIN_0</li> <li>– LL_GPIO_PIN_1</li> <li>– LL_GPIO_PIN_2</li> <li>– LL_GPIO_PIN_3</li> <li>– LL_GPIO_PIN_4</li> <li>– LL_GPIO_PIN_5</li> <li>– LL_GPIO_PIN_6</li> <li>– LL_GPIO_PIN_7</li> <li>– LL_GPIO_PIN_8</li> <li>– LL_GPIO_PIN_9</li> <li>– LL_GPIO_PIN_10</li> <li>– LL_GPIO_PIN_11</li> </ul> </li> </ul> |

---

|                                                   |                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_GPIO_PIN_12</li> <li>- LL_GPIO_PIN_13</li> <li>- LL_GPIO_PIN_14</li> <li>- LL_GPIO_PIN_15</li> <li>- LL_GPIO_PIN_ALL</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IDR IDy LL_GPIO_IsInputPinSet</li> </ul>                                                                                           |

### LL\_GPIO\_WriteOutputPort

|                                                   |                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_GPIO_WriteOutputPort(GPIO_TypeDef * GPIOx, uint32_t PortValue)</code>                                           |
| Function description                              | Write output data register for the port.                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>PortValue:</b> Level value for each pin of the port</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ODR ODy LL_GPIO_WriteOutputPort</li> </ul>                                                           |

### LL\_GPIO\_ReadOutputPort

|                                                   |                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_GPIO_ReadOutputPort(GPIO_TypeDef * GPIOx)</code>             |
| Function description                              | Return full output data register value for a dedicated port.                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> </ul>                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Output:</b> data register value of port</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ODR ODy LL_GPIO_ReadOutputPort</li> </ul>             |

### LL\_GPIO\_IsOutputPinSet

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_GPIO_IsOutputPinSet(GPIO_TypeDef * GPIOx, uint32_t PinMask)</code>                                                                                                                                                                                                                                                                |
| Function description | Return if input data level for several pins of dedicated port is high or low.                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>PinMask:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_GPIO_PIN_0</li> <li>- LL_GPIO_PIN_1</li> <li>- LL_GPIO_PIN_2</li> <li>- LL_GPIO_PIN_3</li> <li>- LL_GPIO_PIN_4</li> <li>- LL_GPIO_PIN_5</li> </ul> </li> </ul> |

- LL\_GPIO\_PIN\_6
- LL\_GPIO\_PIN\_7
- LL\_GPIO\_PIN\_8
- LL\_GPIO\_PIN\_9
- LL\_GPIO\_PIN\_10
- LL\_GPIO\_PIN\_11
- LL\_GPIO\_PIN\_12
- LL\_GPIO\_PIN\_13
- LL\_GPIO\_PIN\_14
- LL\_GPIO\_PIN\_15
- LL\_GPIO\_PIN\_ALL

Return values

- **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ODR ODy LL\_GPIO\_IsOutputPinSet

### LL\_GPIO\_SetOutputPin

Function name

`__STATIC_INLINE void LL_GPIO_SetOutputPin  
(GPIO_TypeDef * GPIOx, uint32_t PinMask)`

Function description

Set several pins to high level on dedicated gpio port.

Parameters

- **GPIOx:** GPIO Port
- **PinMask:** This parameter can be a combination of the following values:

- LL\_GPIO\_PIN\_0
- LL\_GPIO\_PIN\_1
- LL\_GPIO\_PIN\_2
- LL\_GPIO\_PIN\_3
- LL\_GPIO\_PIN\_4
- LL\_GPIO\_PIN\_5
- LL\_GPIO\_PIN\_6
- LL\_GPIO\_PIN\_7
- LL\_GPIO\_PIN\_8
- LL\_GPIO\_PIN\_9
- LL\_GPIO\_PIN\_10
- LL\_GPIO\_PIN\_11
- LL\_GPIO\_PIN\_12
- LL\_GPIO\_PIN\_13
- LL\_GPIO\_PIN\_14
- LL\_GPIO\_PIN\_15
- LL\_GPIO\_PIN\_ALL

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- BSRR BSy LL\_GPIO\_SetOutputPin

**LL\_GPIO\_ResetOutputPin**

Function name      **`_STATIC_INLINE void LL_GPIO_ResetOutputPin  
(GPIO_TypeDef * GPIOx, uint32_t PinMask)`**

Function description      Set several pins to low level on dedicated gpio port.

Parameters     
 

- **GPIOx:** GPIO Port
- **PinMask:** This parameter can be a combination of the following values:

- LL\_GPIO\_PIN\_0
- LL\_GPIO\_PIN\_1
- LL\_GPIO\_PIN\_2
- LL\_GPIO\_PIN\_3
- LL\_GPIO\_PIN\_4
- LL\_GPIO\_PIN\_5
- LL\_GPIO\_PIN\_6
- LL\_GPIO\_PIN\_7
- LL\_GPIO\_PIN\_8
- LL\_GPIO\_PIN\_9
- LL\_GPIO\_PIN\_10
- LL\_GPIO\_PIN\_11
- LL\_GPIO\_PIN\_12
- LL\_GPIO\_PIN\_13
- LL\_GPIO\_PIN\_14
- LL\_GPIO\_PIN\_15
- LL\_GPIO\_PIN\_ALL

Return values     
 

- **None**

Reference Manual to  
LL API cross  
reference:     
 

- BRR BRy LL\_GPIO\_ResetOutputPin

**LL\_GPIO\_TogglePin**

Function name      **`_STATIC_INLINE void LL_GPIO_TogglePin (GPIO_TypeDef *  
GPIOx, uint32_t PinMask)`**

Function description      Toggle data value for several pin of dedicated port.

Parameters     
 

- **GPIOx:** GPIO Port
- **PinMask:** This parameter can be a combination of the following values:

- LL\_GPIO\_PIN\_0
- LL\_GPIO\_PIN\_1
- LL\_GPIO\_PIN\_2
- LL\_GPIO\_PIN\_3
- LL\_GPIO\_PIN\_4
- LL\_GPIO\_PIN\_5
- LL\_GPIO\_PIN\_6
- LL\_GPIO\_PIN\_7
- LL\_GPIO\_PIN\_8
- LL\_GPIO\_PIN\_9
- LL\_GPIO\_PIN\_10
- LL\_GPIO\_PIN\_11

- LL\_GPIO\_PIN\_12
- LL\_GPIO\_PIN\_13
- LL\_GPIO\_PIN\_14
- LL\_GPIO\_PIN\_15
- LL\_GPIO\_PIN\_ALL

**Return values**

- **None**

Reference Manual to  
LL API cross  
reference:

- ODR ODy LL\_GPIO\_TogglePin

### LL\_GPIO\_DeInit

|                      |                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_GPIO_DeInit (GPIO_TypeDef * GPIOx)</b>                                                                                                                                                                                  |
| Function description | De-initialize GPIO registers (Registers restored to their default values).                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> </ul>                                                                                                                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>- SUCCESS: GPIO registers are de-initialized</li> <li>- ERROR: Wrong GPIO Port</li> </ul> </li> </ul> |

### LL\_GPIO\_Init

|                      |                                                                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_GPIO_Init (GPIO_TypeDef * GPIOx,<br/>LL_GPIO_InitTypeDef * GPIO_InitStruct)</b>                                                                                                                                                                          |
| Function description | Initialize GPIO registers according to the specified parameters in GPIO_InitStruct.                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIOx:</b> GPIO Port</li> <li>• <b>GPIO_InitStruct:</b> pointer to a LL_GPIO_InitTypeDef structure that contains the configuration information for the specified GPIO peripheral.</li> </ul>                                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>- SUCCESS: GPIO registers are initialized according to GPIO_InitStruct content</li> <li>- ERROR: Not applicable</li> </ul> </li> </ul> |

### LL\_GPIO\_StructInit

|                      |                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_GPIO_StructInit (LL_GPIO_InitTypeDef *<br/>GPIO_InitStruct)</b>                                                                                         |
| Function description | Set each LL_GPIO_InitTypeDef field to default value.                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>GPIO_InitStruct:</b> pointer to a LL_GPIO_InitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                    |

## 67.3 GPIO Firmware driver defines

### 67.3.1 GPIO

#### *Alternate Function*

|               |                              |
|---------------|------------------------------|
| LL_GPIO_AF_0  | Select alternate function 0  |
| LL_GPIO_AF_1  | Select alternate function 1  |
| LL_GPIO_AF_2  | Select alternate function 2  |
| LL_GPIO_AF_3  | Select alternate function 3  |
| LL_GPIO_AF_4  | Select alternate function 4  |
| LL_GPIO_AF_5  | Select alternate function 5  |
| LL_GPIO_AF_6  | Select alternate function 6  |
| LL_GPIO_AF_7  | Select alternate function 7  |
| LL_GPIO_AF_8  | Select alternate function 8  |
| LL_GPIO_AF_9  | Select alternate function 9  |
| LL_GPIO_AF_10 | Select alternate function 10 |
| LL_GPIO_AF_11 | Select alternate function 11 |
| LL_GPIO_AF_12 | Select alternate function 12 |
| LL_GPIO_AF_13 | Select alternate function 13 |
| LL_GPIO_AF_14 | Select alternate function 14 |
| LL_GPIO_AF_15 | Select alternate function 15 |

#### *Mode*

|                        |                                |
|------------------------|--------------------------------|
| LL_GPIO_MODE_INPUT     | Select input mode              |
| LL_GPIO_MODE_OUTPUT    | Select output mode             |
| LL_GPIO_MODE_ALTERNATE | Select alternate function mode |
| LL_GPIO_MODE_ANALOG    | Select analog mode             |

#### *Output Type*

|                          |                                  |
|--------------------------|----------------------------------|
| LL_GPIO_OUTPUT_PUSHPULL  | Select push-pull as output type  |
| LL_GPIO_OUTPUT_OPENDRAIN | Select open-drain as output type |

#### *PIN*

|               |              |
|---------------|--------------|
| LL_GPIO_PIN_0 | Select pin 0 |
| LL_GPIO_PIN_1 | Select pin 1 |
| LL_GPIO_PIN_2 | Select pin 2 |
| LL_GPIO_PIN_3 | Select pin 3 |
| LL_GPIO_PIN_4 | Select pin 4 |
| LL_GPIO_PIN_5 | Select pin 5 |
| LL_GPIO_PIN_6 | Select pin 6 |
| LL_GPIO_PIN_7 | Select pin 7 |

---

|                 |                 |
|-----------------|-----------------|
| LL_GPIO_PIN_8   | Select pin 8    |
| LL_GPIO_PIN_9   | Select pin 9    |
| LL_GPIO_PIN_10  | Select pin 10   |
| LL_GPIO_PIN_11  | Select pin 11   |
| LL_GPIO_PIN_12  | Select pin 12   |
| LL_GPIO_PIN_13  | Select pin 13   |
| LL_GPIO_PIN_14  | Select pin 14   |
| LL_GPIO_PIN_15  | Select pin 15   |
| LL_GPIO_PIN_ALL | Select all pins |

**Pull Up Pull Down**

|                   |                      |
|-------------------|----------------------|
| LL_GPIO_PULL_NO   | Select I/O no pull   |
| LL_GPIO_PULL_UP   | Select I/O pull up   |
| LL_GPIO_PULL_DOWN | Select I/O pull down |

**Output Speed**

|                           |                                |
|---------------------------|--------------------------------|
| LL_GPIO_SPEED_FREQ_LOW    | Select I/O low output speed    |
| LL_GPIO_SPEED_FREQ_MEDIUM | Select I/O medium output speed |
| LL_GPIO_SPEED_FREQ_HIGH   | Select I/O high output speed   |

**Common Write and read registers Macros**

**LL\_GPIO\_WriteReg**      **Description:**

- Write a value in GPIO register.

**Parameters:**

- `__INSTANCE__`: GPIO Instance
- `__REG__`: Register to be written
- `__VALUE__`: Value to be written in the register

**Return value:**

- None

**LL\_GPIO\_ReadReg**      **Description:**

- Read a value in GPIO register.

**Parameters:**

- `__INSTANCE__`: GPIO Instance
- `__REG__`: Register to be read

**Return value:**

- Register: value

## 68 LL HRTIM Generic Driver

### 68.1 HRTIM Firmware driver API description

#### 68.1.1 Detailed description of functions

##### **LL\_HRTIM\_SetSyncInSrc**

|                                                   |                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_SetSyncInSrc<br/>(HRTIM_TypeDef * HRTIMx, uint32_t SyncInSrc)</code>                                                                                                                                                                                                                                               |
| Function description                              | Select the HRTIM synchronization input source.                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>SyncInSrc:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_SYNCIN_SRC_NONE</li> <li>– LL_HRTIM_SYNCIN_SRC_TIM_EVENT</li> <li>– LL_HRTIM_SYNCIN_SRC_EXTERNAL_EVENT</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the concerned timer(s) is (are) enabled .</li> </ul>                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR SYNCIN LL_HRTIM_SetSyncInSrc</li> </ul>                                                                                                                                                                                                                                                                   |

##### **LL\_HRTIM\_GetSyncInSrc**

|                                                   |                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_GetSyncInSrc<br/>(HRTIM_TypeDef * HRTIMx)</code>                                                                                                                                                                                                      |
| Function description                              | Get actual HRTIM synchronization input source.                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>SyncInSrc:</b> Returned value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_SYNCIN_SRC_NONE</li> <li>– LL_HRTIM_SYNCIN_SRC_TIM_EVENT</li> <li>– LL_HRTIM_SYNCIN_SRC_EXTERNAL_EVENT</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR SYNCIN LL_HRTIM_SetSyncInSrc</li> </ul>                                                                                                                                                                                                          |

##### **LL\_HRTIM\_ConfigSyncOut**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_ConfigSyncOut<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Config, uint32_t Src)</code>                                                           |
| Function description | Configure the HRTIM synchronization output.                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Config:</b> This parameter can be one of the following values:</li> </ul> |

- LL\_HRTIM\_SYNCOUT\_DISABLED
- LL\_HRTIM\_SYNCOUT\_POSITIVE\_PULSE
- LL\_HRTIM\_SYNCOUT\_NEGATIVE\_PULSE
- **Src:** This parameter can be one of the following values:
  - LL\_HRTIM\_SYNCOUT\_SRC\_MASTER\_START
  - LL\_HRTIM\_SYNCOUT\_SRC\_MASTER\_CMP1
  - LL\_HRTIM\_SYNCOUT\_SRC\_TIMA\_START
  - LL\_HRTIM\_SYNCOUT\_SRC\_TIMA\_CMP1

**Return values**

- **None**

**Reference Manual to  
LL API cross  
reference:**

- MCR SYNCSRC LL\_HRTIM\_ConfigSyncOut
- MCR SYNCOUT LL\_HRTIM\_ConfigSyncOut

**LL\_HRTIM\_SetSyncOutConfig****Function name**

```
__STATIC_INLINE void LL_HRTIM_SetSyncOutConfig
(HRTIM_TypeDef * HRTIMx, uint32_t SyncOutConfig)
```

**Function description**

Set the routing and conditioning of the synchronization output event.

**Parameters**

- **HRTIMx:** High Resolution Timer instance
- **SyncOutConfig:** This parameter can be one of the following values:
  - LL\_HRTIM\_SYNCOUT\_DISABLED
  - LL\_HRTIM\_SYNCOUT\_POSITIVE\_PULSE
  - LL\_HRTIM\_SYNCOUT\_NEGATIVE\_PULSE

**Return values**

- **None**

**Notes**

- This function can be called only when the master timer is enabled.

**Reference Manual to  
LL API cross  
reference:**

- MCR SYNCOUT LL\_HRTIM\_SetSyncOutConfig

**LL\_HRTIM\_GetSyncOutConfig****Function name**

```
__STATIC_INLINE uint32_t LL_HRTIM_GetSyncOutConfig
(HRTIM_TypeDef * HRTIMx)
```

**Function description**

Get actual routing and conditioning of the synchronization output event.

**Parameters**

- **HRTIMx:** High Resolution Timer instance

**Return values**

- **SyncOutConfig:** Returned value can be one of the following values:
  - LL\_HRTIM\_SYNCOUT\_DISABLED
  - LL\_HRTIM\_SYNCOUT\_POSITIVE\_PULSE
  - LL\_HRTIM\_SYNCOUT\_NEGATIVE\_PULSE

**Reference Manual to  
LL API cross  
reference:**

- MCR SYNCOUT LL\_HRTIM\_GetSyncOutConfig

**LL\_HRTIM\_SetSyncOutSrc**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_SetSyncOutSrc<br/>(HRTIM_TypeDef * HRTIMx, uint32_t SyncOutSrc)</code>                                                                                                                                                                                                                                                                                                 |
| Function description                              | Set the source and event to be sent on the HRTIM synchronization output.                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>SyncOutSrc:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_SYNCOUT_SRC_MASTER_START</li> <li>– LL_HRTIM_SYNCOUT_SRC_MASTER_CMP1</li> <li>– LL_HRTIM_SYNCOUT_SRC_TIMA_START</li> <li>– LL_HRTIM_SYNCOUT_SRC_TIMA_CMP1</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR SYNCNSRC LL_HRTIM_SetSyncOutSrc</li> </ul>                                                                                                                                                                                                                                                                                                                    |

**LL\_HRTIM\_GetSyncOutSrc**

|                                                   |                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_GetSyncOutSrc<br/>(HRTIM_TypeDef * HRTIMx)</code>                                                                                                                                                                                                                                                         |
| Function description                              | Get actual source and event sent on the HRTIM synchronization output.                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                                                                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>SyncOutSrc:</b> Returned value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_SYNCOUT_SRC_MASTER_START</li> <li>– LL_HRTIM_SYNCOUT_SRC_MASTER_CMP1</li> <li>– LL_HRTIM_SYNCOUT_SRC_TIMA_START</li> <li>– LL_HRTIM_SYNCOUT_SRC_TIMA_CMP1</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR SYNCNSRC LL_HRTIM_GetSyncOutSrc</li> </ul>                                                                                                                                                                                                                                                           |

**LL\_HRTIM\_SuspendUpdate**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_SuspendUpdate<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timers)</code>                                                                                                                                                                                                                                                                                                          |
| Function description | Disable (temporarily) update event generation.                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timers:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>Allow to temporarily disable the transfer from preload to active registers, whatever the selected update event. This allows to modify several registers in multiple timers.</li> </ul>                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 MUDIS LL_HRTIM_SuspendUpdate</li> <li>CR1 TAUDIS LL_HRTIM_SuspendUpdate</li> <li>CR1 TBUDIS LL_HRTIM_SuspendUpdate</li> <li>CR1 TCUDIS LL_HRTIM_SuspendUpdate</li> <li>CR1 TDUDIS LL_HRTIM_SuspendUpdate</li> <li>CR1 TEUDIS LL_HRTIM_SuspendUpdate</li> </ul> |

### LL\_HRTIM\_ResumeUpdate

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_HRTIM_ResumeUpdate(HRTIM_TypeDef * HRTIMx, uint32_t Timers)</code></b>                                                                                                                                                                                                                                                                                                     |
| Function description                              | Enable update event generation.                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timers:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>The regular update event takes place.</li> </ul>                                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 MUDIS LL_HRTIM_ResumeUpdate</li> <li>CR1 TAUDIS LL_HRTIM_ResumeUpdate</li> <li>CR1 TBUDIS LL_HRTIM_ResumeUpdate</li> <li>CR1 TCUDIS LL_HRTIM_ResumeUpdate</li> <li>CR1 TDUDIS LL_HRTIM_ResumeUpdate</li> <li>CR1 TEUDIS LL_HRTIM_ResumeUpdate</li> </ul>                                                                                                         |

### LL\_HRTIM\_ForceUpdate

|                      |                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_HRTIM_ForceUpdate(HRTIM_TypeDef * HRTIMx, uint32_t Timers)</code></b>                                                                                                                                                                                                                                                                          |
| Function description | Force an immediate transfer from the preload to the active register                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timers:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | – LL_HRTIM_TIMER_E                                                                                                                                                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• Any pending update request is cancelled.</li> </ul>                                                                                                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 MSWU LL_HRTIM_ForceUpdate</li> <li>• CR2 TASWU LL_HRTIM_ForceUpdate</li> <li>• CR2 TBSWU LL_HRTIM_ForceUpdate</li> <li>• CR2 TCSWU LL_HRTIM_ForceUpdate</li> <li>• CR2 TDSWU LL_HRTIM_ForceUpdate</li> <li>• CR2 TESWU LL_HRTIM_ForceUpdate</li> </ul> |

### LL\_HRTIM\_CounterReset

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>STATIC_INLINE void LL_HRTIM_CounterReset<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timers)</b>                                                                                                                                                                                                                                                                                                                             |
| Function description                              | Reset the HRTIM timer(s) counter.                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timers:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 MRST LL_HRTIM_CounterReset</li> <li>• CR2 TARST LL_HRTIM_CounterReset</li> <li>• CR2 TBRST LL_HRTIM_CounterReset</li> <li>• CR2 TCRST LL_HRTIM_CounterReset</li> <li>• CR2 TDRST LL_HRTIM_CounterReset</li> <li>• CR2 TERST LL_HRTIM_CounterReset</li> </ul>                                                                                                                 |

### LL\_HRTIM\_EnableOutput

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>STATIC_INLINE void LL_HRTIM_EnableOutput<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Outputs)</b>                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Enable the HRTIM timer(s) output(s) .                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Outputs:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_OUTPUT_TA1</li> <li>– LL_HRTIM_OUTPUT_TA2</li> <li>– LL_HRTIM_OUTPUT_TB1</li> <li>– LL_HRTIM_OUTPUT_TB2</li> <li>– LL_HRTIM_OUTPUT_TC1</li> <li>– LL_HRTIM_OUTPUT_TC2</li> <li>– LL_HRTIM_OUTPUT_TD1</li> <li>– LL_HRTIM_OUTPUT_TD2</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OENR TA1OEN LL_HRTIM_EnableOutput</li> <li>• OENR TA2OEN LL_HRTIM_EnableOutput</li> <li>• OENR TB1OEN LL_HRTIM_EnableOutput</li> <li>• OENR TB2OEN LL_HRTIM_EnableOutput</li> <li>• OENR TC1OEN LL_HRTIM_EnableOutput</li> <li>• OENR TC2OEN LL_HRTIM_EnableOutput</li> <li>• OENR TD1OEN LL_HRTIM_EnableOutput</li> <li>• OENR TD2OEN LL_HRTIM_EnableOutput</li> <li>• OENR TE1OEN LL_HRTIM_EnableOutput</li> <li>• OENR TE2OEN LL_HRTIM_EnableOutput</li> </ul> |

### LL\_HRTIM\_DisableOutput

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>STATIC_INLINE void LL_HRTIM_DisableOutput(<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Outputs)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Disable the HRTIM timer(s) output(s) .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Outputs:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OENR TA1OEN LL_HRTIM_DisableOutput</li> <li>• OENR TA2OEN LL_HRTIM_DisableOutput</li> <li>• OENR TB1OEN LL_HRTIM_DisableOutput</li> <li>• OENR TB2OEN LL_HRTIM_DisableOutput</li> <li>• OENR TC1OEN LL_HRTIM_DisableOutput</li> <li>• OENR TC2OEN LL_HRTIM_DisableOutput</li> <li>• OENR TD1OEN LL_HRTIM_DisableOutput</li> <li>• OENR TD2OEN LL_HRTIM_DisableOutput</li> <li>• OENR TE1OEN LL_HRTIM_DisableOutput</li> <li>• OENR TE2OEN LL_HRTIM_DisableOutput</li> </ul>                                      |

### LL\_HRTIM\_IsEnabledOutput

|                      |                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------|
| Function name        | <b>STATIC_INLINE uint32_t LL_HRTIM_IsEnabledOutput(<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Output)</b> |
| Function description | Indicates whether the HRTIM timer output is enabled.                                                  |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Output:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of TxyOEN bit in HRTIM_OENR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>OENR TA1OEN LL_HRTIM_IsEnabledOutput</li> <li>OENR TA2OEN LL_HRTIM_IsEnabledOutput</li> <li>OENR TB1OEN LL_HRTIM_IsEnabledOutput</li> <li>OENR TB2OEN LL_HRTIM_IsEnabledOutput</li> <li>OENR TC1OEN LL_HRTIM_IsEnabledOutput</li> <li>OENR TC2OEN LL_HRTIM_IsEnabledOutput</li> <li>OENR TD1OEN LL_HRTIM_IsEnabledOutput</li> <li>OENR TD2OEN LL_HRTIM_IsEnabledOutput</li> <li>OENR TE1OEN LL_HRTIM_IsEnabledOutput</li> <li>OENR TE2OEN LL_HRTIM_IsEnabledOutput</li> </ul>                                 |

## LL\_HRTIM\_IsDisabledOutput

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsDisabledOutput(HRTIM_TypeDef * HRTIMx, uint32_t Output)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description                              | Indicates whether the HRTIM timer output is disabled.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Output:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of TxyODS bit in HRTIM_ODSR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ODISR TA1ODIS LL_HRTIM_IsDisabledOutput</li> <li>ODISR TA2ODIS LL_HRTIM_IsDisabledOutput</li> <li>ODISR TB1ODIS LL_HRTIM_IsDisabledOutput</li> <li>ODISR TB2ODIS LL_HRTIM_IsDisabledOutput</li> <li>ODISR TC1ODIS LL_HRTIM_IsDisabledOutput</li> <li>ODISR TC2ODIS LL_HRTIM_IsDisabledOutput</li> <li>ODISR TD1ODIS LL_HRTIM_IsDisabledOutput</li> </ul>                                                                                                                                                      |

- ODISR TD2ODIS LL\_HRTIM\_IsDisabledOutput
- ODISR TE1ODIS LL\_HRTIM\_IsDisabledOutput
- ODISR TE2ODIS LL\_HRTIM\_IsDisabledOutput

### LL\_HRTIM\_ConfigADCTrig

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_HRTIM_ConfigADCTrig(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t ADCTrig, uint32_t<br/>    Update, uint32_t Src)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                        | Configure an ADC trigger.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>ADCTrig:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_ADCTRIG_1</li> <li>- LL_HRTIM_ADCTRIG_2</li> <li>- LL_HRTIM_ADCTRIG_3</li> <li>- LL_HRTIM_ADCTRIG_4</li> </ul> </li> <li>• <b>Update:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_ADCTRIG_UPDATE_MASTER</li> <li>- LL_HRTIM_ADCTRIG_UPDATE_TIMER_A</li> <li>- LL_HRTIM_ADCTRIG_UPDATE_TIMER_B</li> <li>- LL_HRTIM_ADCTRIG_UPDATE_TIMER_C</li> <li>- LL_HRTIM_ADCTRIG_UPDATE_TIMER_D</li> <li>- LL_HRTIM_ADCTRIG_UPDATE_TIMER_E</li> </ul> </li> <li>• <b>Src:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>• CR1 ADC1USRC LL_HRTIM_ConfigADCTrig</li> <li>• CR1 ADC2USRC LL_HRTIM_ConfigADCTrig</li> <li>• CR1 ADC3USRC LL_HRTIM_ConfigADCTrig</li> <li>• CR1 ADC4USRC LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1MC4 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1MPER LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1EEV1 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1EEV2 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1EEV3 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1EEV4 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1EEV5 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TAC2 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TAC3 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TAC4 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TAPER LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TARST LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TBC2 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TBC3 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TBC4 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TBPER LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TBRST LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TCC2 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TCC3 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TCC4 LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TCPER LL_HRTIM_ConfigADCTrig</li> <li>• ADC1R ADC1TDC2 LL_HRTIM_ConfigADCTrig</li> </ul> </li> </ul> |
| Reference Manual to LL API cross reference: |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

- ADC1R ADC1TDC3 LL\_HRTIM\_ConfigADCTrig
- ADC1R ADC1TDC4 LL\_HRTIM\_ConfigADCTrig
- ADC1R ADC1TDPER LL\_HRTIM\_ConfigADCTrig
- ADC1R ADC1TEC2 LL\_HRTIM\_ConfigADCTrig
- ADC1R ADC1TEC3 LL\_HRTIM\_ConfigADCTrig
- ADC1R ADC1TEC4 LL\_HRTIM\_ConfigADCTrig
- ADC1R ADC1TEPER LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2MC1 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2MC2 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2MC3 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2MC4 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2MPER LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2EEV6 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2EEV7 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2EEV8 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2EEV9 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2EEV10 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TAC2 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TAC3 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TAC4 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TAPER LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TBC2 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TBC3 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TBC4 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TBPER LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TCC2 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TCC3 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TCC4 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TCPER LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TCRST LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TDC2 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TDC3 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TDC4 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TDPER LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TDRST LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TEC2 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TEC3 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TEC4 LL\_HRTIM\_ConfigADCTrig
- ADC2R ADC2TERST LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3MC1 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3MC2 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3MC3 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3MC4 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3MPER LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3EEV1 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3EEV2 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3EEV3 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3EEV4 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3EEV5 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TAC2 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TAC3 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TAC4 LL\_HRTIM\_ConfigADCTrig

- ADC3R ADC3TAPER LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TARST LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TBC2 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TBC3 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TBC4 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TBPER LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TBRST LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TCC2 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TCC3 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TCC4 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TCPER LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TDC2 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TDC3 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TDC4 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TDPER LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TEC2 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TEC3 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TEC4 LL\_HRTIM\_ConfigADCTrig
- ADC3R ADC3TEPER LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4MC1 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4MC2 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4MC3 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4MC4 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4MPER LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4EEV6 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4EEV7 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4EEV8 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4EEV9 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4EEV10 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TAC2 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TAC3 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TAC4 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TAPER LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TBC2 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TBC3 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TBC4 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TBPER LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TCC2 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TCC3 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TCC4 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TCPER LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TCRST LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TDC2 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TDC3 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TDC4 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TDPER LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TDRST LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TEC2 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TEC3 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TEC4 LL\_HRTIM\_ConfigADCTrig
- ADC4R ADC4TERST LL\_HRTIM\_ConfigADCTrig

**LL\_HRTIM\_SetADCTrigUpdate**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_SetADCTrigUpdate(HRTIM_TypeDef * HRTIMx, uint32_t ADCTrig, uint32_t Update)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description                              | Associate the ADCx trigger to a timer triggering the update of the HRTIM_ADCxR register.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>ADCTrig:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_ADCTRIG_1</li> <li>– LL_HRTIM_ADCTRIG_2</li> <li>– LL_HRTIM_ADCTRIG_3</li> <li>– LL_HRTIM_ADCTRIG_4</li> </ul> </li> <li>• <b>Update:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_ADCTRIG_UPDATE_MASTER</li> <li>– LL_HRTIM_ADCTRIG_UPDATE_TIMER_A</li> <li>– LL_HRTIM_ADCTRIG_UPDATE_TIMER_B</li> <li>– LL_HRTIM_ADCTRIG_UPDATE_TIMER_C</li> <li>– LL_HRTIM_ADCTRIG_UPDATE_TIMER_D</li> <li>– LL_HRTIM_ADCTRIG_UPDATE_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• When the preload is disabled in the source timer, the HRTIM_ADCxR registers are not preloaded either: a write access will result in an immediate update of the trigger source.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 ADC1USRC LL_HRTIM_SetADCTrigUpdate</li> <li>• CR1 ADC2USRC LL_HRTIM_SetADCTrigUpdate</li> <li>• CR1 ADC3USRC LL_HRTIM_SetADCTrigUpdate</li> <li>• CR1 ADC4USRC LL_HRTIM_SetADCTrigUpdate</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

**LL\_HRTIM\_GetADCTrigUpdate**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_GetADCTrigUpdate(HRTIM_TypeDef * HRTIMx, uint32_t ADCTrig)</code>                                                                                                                                                                                                                                                                                                                        |
| Function description | Get the source timer triggering the update of the HRTIM_ADCxR register.                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>ADCTrig:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_ADCTRIG_1</li> <li>– LL_HRTIM_ADCTRIG_2</li> <li>– LL_HRTIM_ADCTRIG_3</li> <li>– LL_HRTIM_ADCTRIG_4</li> </ul> </li> </ul>                                                                                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Update:</b> Returned value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_ADCTRIG_UPDATE_MASTER</li> <li>– LL_HRTIM_ADCTRIG_UPDATE_TIMER_A</li> <li>– LL_HRTIM_ADCTRIG_UPDATE_TIMER_B</li> <li>– LL_HRTIM_ADCTRIG_UPDATE_TIMER_C</li> <li>– LL_HRTIM_ADCTRIG_UPDATE_TIMER_D</li> <li>– LL_HRTIM_ADCTRIG_UPDATE_TIMER_E</li> </ul> </li> </ul> |

- Reference Manual to  
LL API cross  
reference:
- CR1 ADC1USRC LL\_HRTIM\_SetADCTrigUpdate
  - CR1 ADC2USRC LL\_HRTIM\_SetADCTrigUpdate
  - CR1 ADC3USRC LL\_HRTIM\_SetADCTrigUpdate
  - CR1 ADC4USRC LL\_HRTIM\_SetADCTrigUpdate

### **LL\_HRTIM\_SetADCTrigSrc**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>STATIC_INLINE void LL_HRTIM_SetADCTrigSrc<br/>(HRTIM_TypeDef * HRTIMx, uint32_t ADCTrig, uint32_t Src)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description                              | Specify which events (timer events and/or external events) are used as triggers for ADC conversion.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>ADCTrig:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_ADCTRIG_1</li> <li>- LL_HRTIM_ADCTRIG_2</li> <li>- LL_HRTIM_ADCTRIG_3</li> <li>- LL_HRTIM_ADCTRIG_4</li> </ul> </li> <li>• <b>Src:</b> This parameter can be a combination of the following values:</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ADC1R ADC1MC4 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1MPER LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1EEV1 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1EEV2 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1EEV3 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1EEV4 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1EEV5 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TAC2 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TAC3 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TAC4 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TAPER LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TARST LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TBC2 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TBC3 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TBC4 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TBPER LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TBRST LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TCC2 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TCC3 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TCC4 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TCPER LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TDC2 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TDC3 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TDC4 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TDPER LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TEC2 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TEC3 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TEC4 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC1R ADC1TEPER LL_HRTIM_SetADCTrigSrc</li> <li>• ADC2R ADC2MC1 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC2R ADC2MC2 LL_HRTIM_SetADCTrigSrc</li> <li>• ADC2R ADC2MC3 LL_HRTIM_SetADCTrigSrc</li> </ul> |

- ADC2R ADC2MC4 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2MPER LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2EEV6 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2EEV7 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2EEV8 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2EEV9 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2EEV10 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TAC2 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TAC3 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TAC4 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TAPER LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TBC2 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TBC3 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TBC4 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TBPER LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TCC2 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TCC3 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TCC4 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TCPER LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TCRST LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TDC2 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TDC3 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TDC4 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TDPER LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TDRST LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TEC2 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TEC3 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TEC4 LL\_HRTIM\_SetADCTrigSrc
- ADC2R ADC2TERST LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3MC1 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3MC2 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3MC3 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3MC4 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3MPER LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3EEV1 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3EEV2 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3EEV3 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3EEV4 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3EEV5 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TAC2 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TAC3 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TAC4 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TAPER LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TARST LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TBC2 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TBC3 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TBC4 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TBPER LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TBRST LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TCC2 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TCC3 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TCC4 LL\_HRTIM\_SetADCTrigSrc

- ADC3R ADC3TCPER LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TDC2 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TDC3 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TDC4 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TDPER LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TEC2 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TEC3 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TEC4 LL\_HRTIM\_SetADCTrigSrc
- ADC3R ADC3TEPER LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4MC1 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4MC2 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4MC3 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4MC4 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4MPER LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4EEV6 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4EEV7 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4EEV8 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4EEV9 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4EEV10 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TAC2 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TAC3 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TAC4 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TAPER LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TBC2 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TBC3 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TBC4 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TBPER LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TCC2 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TCC3 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TCC4 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TCPER LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TCRST LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TDC2 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TDC3 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TDC4 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TDPER LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TDRST LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TEC2 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TEC3 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TEC4 LL\_HRTIM\_SetADCTrigSrc
- ADC4R ADC4TERST LL\_HRTIM\_SetADCTrigSrc

### LL\_HRTIM\_GetADCTrigSrc

Function name **`_STATIC_INLINE uint32_t LL_HRTIM_GetADCTrigSrc(  
HRTIM_TypeDef * HRTIMx, uint32_t ADCTrig)`**

Function description Indicate which events (timer events and/or external events) are currently used as triggers for ADC conversion.

Parameters
 

- **HRTIMx:** High Resolution Timer instance
- **ADCTrig:** This parameter can be one of the following values:
  - `LL_HRTIM_ADCTRIG_1`
  - `LL_HRTIM_ADCTRIG_2`

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"><li>– LL_HRTIM_ADCTRIG_3</li><li>– LL_HRTIM_ADCTRIG_4</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• <b>Src:</b> This parameter can be a combination of the following values:<ul style="list-style-type: none"><li>• ADC1R ADC1MC4 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1MPER LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1EEV1 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1EEV2 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1EEV3 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1EEV4 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1EEV5 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TAC2 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TAC3 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TAC4 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TAPER LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TARST LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TBC2 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TBC3 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TBC4 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TBPER LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TBRST LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TCC2 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TCC3 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TCC4 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TCPER LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TDC2 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TDC3 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TDC4 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TDPER LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TEC2 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TEC3 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TEC4 LL_HRTIM_GetADCTrigSrc</li><li>• ADC1R ADC1TEPER LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2MC1 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2MC2 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2MC3 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2MC4 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2MPER LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2EEV6 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2EEV7 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2EEV8 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2EEV9 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2EEV10 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2TAC2 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2TAC3 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2TAPER LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2TBC2 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2TBC3 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2TBC4 LL_HRTIM_GetADCTrigSrc</li><li>• ADC2R ADC2TBPER LL_HRTIM_GetADCTrigSrc</li></ul></li></ul> |

- ADC2R ADC2TCC2 LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TCC3 LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TCC4 LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TCPER LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TCRST LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TDC2 LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TDC3 LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TDC4 LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TDPER LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TDRST LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TEC2 LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TEC3 LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TEC4 LL\_HRTIM\_GetADCTrigSrc
- ADC2R ADC2TERST LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3MC1 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3MC2 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3MC3 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3MC4 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3MPER LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3EEV1 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3EEV2 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3EEV3 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3EEV4 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3EEV5 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TAC2 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TAC3 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TAC4 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TAPER LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3STARST LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TBC2 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TBC3 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TBC4 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TBPER LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TBRST LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TCC2 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TCC3 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TCC4 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TCPER LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TDC2 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TDC3 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TDC4 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TDPER LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TEC2 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TEC3 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TEC4 LL\_HRTIM\_GetADCTrigSrc
- ADC3R ADC3TEPER LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4MC1 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4MC2 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4MC3 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4MC4 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4MPER LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4EEV6 LL\_HRTIM\_GetADCTrigSrc

- ADC4R ADC4EEV7 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4EEV8 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4EEV9 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4EEV10 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TAC2 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TAC3 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TAC4 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TAPER LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TBC2 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TBC3 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TBC4 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TBPER LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TCC2 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TCC3 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TCC4 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TCPER LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TCRST LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TDC2 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TDC3 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TDC4 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TDPER LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TDRST LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TEC2 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TEC3 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TEC4 LL\_HRTIM\_GetADCTrigSrc
- ADC4R ADC4TERST LL\_HRTIM\_GetADCTrigSrc

### LL\_HRTIM\_ConfigDLLCalibration

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_HRTIM_ConfigDLLCalibration(HRTIM_TypeDef * HRTIMx, uint32_t Mode, uint32_t Period)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description                              | Configure the DLL calibration mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Mode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_DLLCALIBRATION_MODE_SINGLESHOT</li> <li>- LL_HRTIM_DLLCALIBRATION_MODE_CONTINUOUS</li> </ul> </li> <li>• <b>Period:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_DLLCALIBRATION_RATE_7300</li> <li>- LL_HRTIM_DLLCALIBRATION_RATE_910</li> <li>- LL_HRTIM_DLLCALIBRATION_RATE_114</li> <li>- LL_HRTIM_DLLCALIBRATION_RATE_14</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Reference<br>Manual to LL API<br>cross reference: | <ul style="list-style-type: none"> <li>• DLLCR CALEN LL_HRTIM_ConfigDLLCalibration</li> <li>• DLLCR CALRTE LL_HRTIM_ConfigDLLCalibration</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

**LL\_HRTIM\_StartDLLCalibration**

|                                                   |                                                                                                                      |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_HRTIM_StartDLLCalibration(<br/>                  HRTIM_TypeDef * HRTIMx)</code></b> |
| Function description                              | Launch DLL calibration.                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DLLCR CAL LL_HRTIM_StartDLLCalibration</li> </ul>                           |

**LL\_HRTIM\_TIM\_CounterEnable**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_HRTIM_TIM_CounterEnable(<br/>                  HRTIM_TypeDef * HRTIMx, uint32_t Timers)</code></b>                                                                                                                                                                                                                                                                             |
| Function description                              | Enable timer(s) counter.                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timers:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER TECEN LL_HRTIM_TIM_CounterEnable</li> <li>• MDIER TDCEN LL_HRTIM_TIM_CounterEnable</li> <li>• MDIER TCCEN LL_HRTIM_TIM_CounterEnable</li> <li>• MDIER TBCEN LL_HRTIM_TIM_CounterEnable</li> <li>• MDIER TACEN LL_HRTIM_TIM_CounterEnable</li> <li>• MDIER MCEN LL_HRTIM_TIM_CounterEnable</li> </ul>                                                             |

**LL\_HRTIM\_TIM\_CounterDisable**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_HRTIM_TIM_CounterDisable(<br/>                  HRTIM_TypeDef * HRTIMx, uint32_t Timers)</code></b>                                                                                                                                                                                                                                                                            |
| Function description | Disable timer(s) counter.                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timers:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                 |

- Reference Manual to LL API cross reference:
- MDIER TECEN LL\_HRTIM\_TIM\_CounterDisable
  - MDIER TDCEN LL\_HRTIM\_TIM\_CounterDisable
  - MDIER TCCEN LL\_HRTIM\_TIM\_CounterDisable
  - MDIER TBCEN LL\_HRTIM\_TIM\_CounterDisable
  - MDIER TACEN LL\_HRTIM\_TIM\_CounterDisable
  - MDIER MCEN LL\_HRTIM\_TIM\_CounterDisable

### **LL\_HRTIM\_TIM\_IsCounterEnabled**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_TIM_IsCounterEnabled(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                    |
| Function description                        | Indicate whether the timer counter is enabled.                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCEN or TxCEN bit HRTIM_MCR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MDIER TECEN LL_HRTIM_TIM_IsCounterEnabled</li> <li>• MDIER TDCEN LL_HRTIM_TIM_IsCounterEnabled</li> <li>• MDIER TCCEN LL_HRTIM_TIM_IsCounterEnabled</li> <li>• MDIER TBCEN LL_HRTIM_TIM_IsCounterEnabled</li> <li>• MDIER TACEN LL_HRTIM_TIM_IsCounterEnabled</li> <li>• MDIER MCEN LL_HRTIM_TIM_IsCounterEnabled</li> </ul>                                |

### **LL\_HRTIM\_TIM\_SetPrescaler**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_HRTIM_TIM_SetPrescaler(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Prescaler)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description | Set the timer clock prescaler ratio.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Prescaler:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_PRESCALERRATIO_MUL32</li> <li>- LL_HRTIM_PRESCALERRATIO_MUL16</li> <li>- LL_HRTIM_PRESCALERRATIO_MUL8</li> <li>- LL_HRTIM_PRESCALERRATIO_MUL4</li> <li>- LL_HRTIM_PRESCALERRATIO_MUL2</li> <li>- LL_HRTIM_PRESCALERRATIO_DIV1</li> <li>- LL_HRTIM_PRESCALERRATIO_DIV2</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | – LL_HRTIM_PRESCALERRATIO_DIV4                                                                                                                                                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>None</li> </ul>                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>The counter clock equivalent frequency (CK_CNT) is equal to <math>f_{HRCK} / 2^{CKPSC[2:0]}</math>.</li> <li>The prescaling ratio cannot be modified once the timer counter is enabled.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>MCR CKPSC LL_HRTIM_TIM_SetPrescaler</li> <li>TIMxCR CKPSC LL_HRTIM_TIM_SetPrescaler</li> </ul>                                                                                                     |

### LL\_HRTIM\_TIM\_GetPrescaler

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE uint32_t LL_HRTIM_TIM_GetPrescaler(<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</b>                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description                              | Get the timer clock prescaler ratio.                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul>                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Prescaler:</b> Returned value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_PRESCALERRATIO_MUL32</li> <li>– LL_HRTIM_PRESCALERRATIO_MUL16</li> <li>– LL_HRTIM_PRESCALERRATIO_MUL8</li> <li>– LL_HRTIM_PRESCALERRATIO_MUL4</li> <li>– LL_HRTIM_PRESCALERRATIO_MUL2</li> <li>– LL_HRTIM_PRESCALERRATIO_DIV1</li> <li>– LL_HRTIM_PRESCALERRATIO_DIV2</li> <li>– LL_HRTIM_PRESCALERRATIO_DIV4</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>MCR CKPSC LL_HRTIM_TIM_SetPrescaler</li> <li>TIMxCR CKPSC LL_HRTIM_TIM_SetPrescaler</li> </ul>                                                                                                                                                                                                                                                                                                                                                              |

### LL\_HRTIM\_TIM\_SetCounterMode

|                      |                                                                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE void LL_HRTIM_TIM_SetCounterMode(<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Mode)</b>                                                                                                                                                                                                                      |
| Function description | Set the counter operating mode mode (single-shot, continuous or re-triggerable).                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul>                                                                                                                                                                                       |
|                                                   | <ul style="list-style-type: none"> <li>• <b>Mode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_MODE_CONTINUOUS</li> <li>- LL_HRTIM_MODE_SINGLESHOT</li> <li>- LL_HRTIM_MODE_RETRIGGERABLE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR CONT LL_HRTIM_TIM_SetCounterMode</li> <li>• MCR RETRIG LL_HRTIM_TIM_SetCounterMode</li> <li>• TIMxCR CONT LL_HRTIM_TIM_SetCounterMode</li> <li>• TIMxCR RETRIG LL_HRTIM_TIM_SetCounterMode</li> </ul>                                     |

### LL\_HRTIM\_TIM\_GetCounterMode

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetCounterMode(<br/>  HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                               |
| Function description                              | Get the counter operating mode mode.                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Mode:</b> Returned value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_MODE_CONTINUOUS</li> <li>- LL_HRTIM_MODE_SINGLESHOT</li> <li>- LL_HRTIM_MODE_RETRIGGERABLE</li> </ul> </li> </ul>                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR CONT LL_HRTIM_TIM_GetCounterMode</li> <li>• MCR RETRIG LL_HRTIM_TIM_GetCounterMode</li> <li>• TIMxCR CONT LL_HRTIM_TIM_GetCounterMode</li> <li>• TIMxCR RETRIG LL_HRTIM_TIM_GetCounterMode</li> </ul>                                                                                                                                                             |

### LL\_HRTIM\_TIM\_EnableHalfMode

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_TIM_EnableHalfMode(<br/>  HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                                   |
| Function description | Enable the half duty-cycle mode.                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                |

|                                             |                                                                                                                                                                                                                                                              |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>When the half mode is enabled, HRTIM_MCMP1R (or HRTIM_CMP1xR) active register is automatically updated with HRTIM_MP PER/2 (or HRTIM_PERxR/2) value when HRTIM_MP ER (or HRTIM_PERxR) register is written.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>MCR HALF LL_HRTIM_TIM_EnableHalfMode</li> <li>TIMxCR HALF LL_HRTIM_TIM_EnableHalfMode</li> </ul>                                                                                                                      |

### LL\_HRTIM\_TIM\_DisableHalfMode

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_HRTIM_TIM_DisableHalfMode (HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                          |
| Function description                        | Disable the half duty-cycle mode.                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                    |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>MCR HALF LL_HRTIM_TIM_DisableHalfMode</li> <li>TIMxCR HALF LL_HRTIM_TIM_DisableHalfMode</li> </ul>                                                                                                                                                                                                                                                        |

### LL\_HRTIM\_TIM\_IsEnabledHalfMode

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_IsEnabledHalfMode (HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                    |
| Function description                        | Indicate whether half duty-cycle mode is enabled for a given timer.                                                                                                                                                                                                                                                                                                                              |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of HALF bit to 1 in HRTIM_MCR or HRTIM_TIMxCR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                 |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>MCR HALF LL_HRTIM_TIM_IsEnabledHalfMode</li> <li>TIMxCR HALF LL_HRTIM_TIM_IsEnabledHalfMode</li> </ul>                                                                                                                                                                                                                                                    |

**LL\_HRTIM\_TIM\_EnableStartOnSync**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_EnableStartOnSync<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                        |
| Function description                              | Enable the timer start when receiving a synchronization input event.                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR SYNCSTRTM LL_HRTIM_TIM_EnableStartOnSync</li> <li>• TIMxCR SYNSTRTA LL_HRTIM_TIM_EnableStartOnSync</li> </ul>                                                                                                                                                                                                                                           |

**LL\_HRTIM\_TIM\_DisableStartOnSync**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_DisableStartOnSync<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                       |
| Function description                              | Disable the timer start when receiving a synchronization input event.                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR SYNCSTRTM LL_HRTIM_TIM_DisableStartOnSync</li> <li>• TIMxCR SYNSTRTA LL_HRTIM_TIM_DisableStartOnSync</li> </ul>                                                                                                                                                                                                                                         |

**LL\_HRTIM\_TIM\_IsEnabledStartOnSync**

|                      |                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_TIM_IsEnabledStartOnSync (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code>                                                                                                                                                             |
| Function description | Indicate whether the timer start when receiving a synchronization input event.                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> </ul> </li> </ul> |

|                                             |                                                                                                                                                                  |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                             | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul>         |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of SYNCSTRTx bit in HRTIM_MCR or HRTIM_TIMxCR register (1 or 0).</li> </ul>                               |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MCR SYNCSTRTM LL_HRTIM_TIM_IsEnabledStartOnSync</li> <li>• TIMxCR SYNSTRTA LL_HRTIM_TIM_IsEnabledStartOnSync</li> </ul> |

### LL\_HRTIM\_TIM\_EnableResetOnSync

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_HRTIM_TIM_EnableResetOnSync(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                             |
| Function description                        | Enable the timer reset when receiving a synchronization input event.                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MCR SYNCRSTM LL_HRTIM_TIM_EnableResetOnSync</li> <li>• TIMxCR SYNCRSTA LL_HRTIM_TIM_EnableResetOnSync</li> </ul>                                                                                                                                                                                                                                            |

### LL\_HRTIM\_TIM\_DisableResetOnSync

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_HRTIM_TIM_DisableResetOnSync(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                            |
| Function description                        | Disable the timer reset when receiving a synchronization input event.                                                                                                                                                                                                                                                                                                                                |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MCR SYNCRSTM LL_HRTIM_TIM_DisableResetOnSync</li> <li>• TIMxCR SYNCRSTA LL_HRTIM_TIM_DisableResetOnSync</li> </ul>                                                                                                                                                                                                                                          |

**LL\_HRTIM\_TIM\_IsEnabledResetOnSync**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_TIM_IsEnabledResetOnSync (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                             |
| Function description                              | Indicate whether the timer reset when receiving a synchronization input event.                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR SYNCRSTM LL_HRTIM_TIM_IsEnabledResetOnSync</li> <li>• TIMxCR SYNCRSTA<br/>LL_HRTIM_TIM_IsEnabledResetOnSync</li> </ul>                                                                                                                                                                                                                                  |

**LL\_HRTIM\_TIM\_SetDACTrig**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_SetDACTrig<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t DACTrig)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description                              | Set the HRTIM output the DAC synchronization event is generated on (DACtrigOutx).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>DACTrig:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_DACTRIG_NONE</li> <li>- LL_HRTIM_DACTRIG_DACTRIGOUT_1</li> <li>- LL_HRTIM_DACTRIG_DACTRIGOUT_2</li> <li>- LL_HRTIM_DACTRIG_DACTRIGOUT_3</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR DACSYNC LL_HRTIM_TIM_SetDACTrig</li> <li>• TIMxCR DACSYNC LL_HRTIM_TIM_SetDACTrig</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

**LL\_HRTIM\_TIM\_GetDACTrig**

|                      |                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetDACTrig<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code> |
| Function description | Get the HRTIM output the DAC synchronization event is generated                                            |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | on (DACtrigOutx).                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>DACTrig:</b> Returned value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_DACTRIG_NONE</li> <li>– LL_HRTIM_DACTRIG_DACTRIGOUT_1</li> <li>– LL_HRTIM_DACTRIG_DACTRIGOUT_2</li> <li>– LL_HRTIM_DACTRIG_DACTRIGOUT_3</li> </ul> </li> </ul>                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR DACSYNC LL_HRTIM_TIM_GetDACTrig</li> <li>• TIMxCR DACSYNC LL_HRTIM_TIM_GetDACTrig</li> </ul>                                                                                                                                                                                                                                                            |

### LL\_HRTIM\_TIM\_EnablePreload

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_TIM_EnablePreload(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                           |
| Function description                              | Enable the timer registers preload mechanism.                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• When the preload mode is enabled, accessed registers are shadow registers. Their content is transferred into the active register after an update request, either software or synchronized with an event.</li> </ul>                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR PREEN LL_HRTIM_TIM_EnablePreload</li> <li>• TIMxCR PREEN LL_HRTIM_TIM_EnablePreload</li> </ul>                                                                                                                                                                                                                                                          |

**LL\_HRTIM\_TIM\_DisablePreload**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_TIM_DisablePreload(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                          |
| Function description                              | Disable the timer registers preload mechanism.                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR PREEN LL_HRTIM_TIM_DisablePreload</li> <li>• TIMxCR PREEN LL_HRTIM_TIM_DisablePreload</li> </ul>                                                                                                                                                                                                                                                        |

**LL\_HRTIM\_TIM\_IsEnabledPreload**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_TIM_IsEnabledPreload(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                    |
| Function description                              | Indicate whether the timer registers preload mechanism is enabled.                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of PREEN bit in HRTIM_MCR or HRTIM_TIMxCR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR PREEN LL_HRTIM_TIM_IsEnabledPreload</li> <li>• TIMxCR PREEN LL_HRTIM_TIM_IsEnabledPreload</li> </ul>                                                                                                                                                                                                                                                    |

**LL\_HRTIM\_TIM\_SetUpdateTrig**

|                      |                                                                                                                                                                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_HRTIM_TIM_SetUpdateTrig(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t UpdateTrig)</code></b>                                                                                                                                                                                  |
| Function description | Set the timer register update trigger.                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> </ul> </li> </ul> |

- LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E
  - **UpdateTrig:** This parameter can be one of the following values:
    - MCR MREPU LL\_HRTIM\_TIM\_SetUpdateTrig
    - TIMxCR TAU LL\_HRTIM\_TIM\_SetUpdateTrig
    - TIMxCR TBU LL\_HRTIM\_TIM\_SetUpdateTrig
    - TIMxCR TCU LL\_HRTIM\_TIM\_SetUpdateTrig
    - TIMxCR TDU LL\_HRTIM\_TIM\_SetUpdateTrig
    - TIMxCR TEU LL\_HRTIM\_TIM\_SetUpdateTrig
    - TIMxCR MSTU LL\_HRTIM\_TIM\_SetUpdateTrig
- Reference Manual to LL API cross reference:**

### LL\_HRTIM\_TIM\_GetUpdateTrig

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetUpdateTrig(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                        | Set the timer register update trigger.                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul>                                                                      |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>UpdateTrig:</b> Returned value can be one of the following values:           <ul style="list-style-type: none"> <li>• MCR MREPU LL_HRTIM_TIM_GetUpdateTrig</li> <li>• TIMxCR TBU LL_HRTIM_TIM_GetUpdateTrig</li> <li>• TIMxCR TCU LL_HRTIM_TIM_GetUpdateTrig</li> <li>• TIMxCR TDU LL_HRTIM_TIM_GetUpdateTrig</li> <li>• TIMxCR TEU LL_HRTIM_TIM_GetUpdateTrig</li> <li>• TIMxCR MSTU LL_HRTIM_TIM_GetUpdateTrig</li> </ul> </li> </ul> |
| Reference Manual to LL API cross reference: |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

### LL\_HRTIM\_TIM\_SetUpdateGating

|                      |                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_TIM_SetUpdateGating(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t UpdateGating)</code>                                                                                                                                                                                                                          |
| Function description | Set the timer registers update condition (how the registers update occurs relatively to the burst DMA transaction or an external update request received on one of the update enable inputs (UPD_EN[3:1])).                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> </ul> </li> </ul> |

- LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E
  - **UpdateGating:** This parameter can be one of the following values:
    - MCR BRSTDMA LL\_HRTIM\_TIM\_SetUpdateGating
    - TIMxCR UPDGAT LL\_HRTIM\_TIM\_SetUpdateGating
- Reference Manual to  
LL API cross  
reference:

### **LL\_HRTIM\_TIM\_GetUpdateGating**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetUpdateGating(<br/>          HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                      |
| Function description                              | Get the timer registers update condition.                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>UpdateGating:</b> Returned value can be one of the following values:</li> </ul>                                                                                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCR BRSTDMA LL_HRTIM_TIM_GetUpdateGating</li> <li>• TIMxCR UPDGAT LL_HRTIM_TIM_GetUpdateGating</li> </ul>                                                                                                                                                                                                                                                             |

### **LL\_HRTIM\_TIM\_EnablePushPullMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_EnablePushPullMode(<br/>          HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                      |
| Function description                              | Enable the push-pull mode.                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxCR PSHPLL LL_HRTIM_TIM_EnablePushPullMode</li> </ul>                                                                                                                                                                                                                                                                             |

**LL\_HRTIM\_TIM\_DisablePushPullMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_DisablePushPullMode<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                     |
| Function description                              | Disable the push-pull mode.                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxCR PSHPLL LL_HRTIM_TIM_DisablePushPullMode</li> </ul>                                                                                                                                                                                                                                                                  |

**LL\_HRTIM\_TIM\_IsEnabledPushPullMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_TIM_IsEnabledPushPullMode (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                           |
| Function description                              | Indicate whether the push-pull mode is enabled.                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of PSHPLL bit in HRTIM_TIMxCR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxCR PSHPLL LL_HRTIM_TIM_IsEnabledPushPullMode</li> </ul>                                                                                                                                                                                                                                                                |

**LL\_HRTIM\_TIM\_SetCompareMode**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_TIM_SetCompareMode<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t<br/>CompareUnit, uint32_t Mode)</code>                                                                                                                                                                                                                 |
| Function description | Set the functioning mode of the compare unit (CMP2 or CMP4 can operate in standard mode or in auto delayed mode).                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

- **CompareUnit:** This parameter can be one of the following values:
    - LL\_HRTIM\_COMPAREUNIT\_2
    - LL\_HRTIM\_COMPAREUNIT\_4
  - **Mode:** This parameter can be one of the following values:
    - LL\_HRTIM\_COMPAREMODE\_REGULAR
    - LL\_HRTIM\_COMPAREMODE\_DELAY\_NOTIMEOUT
    - LL\_HRTIM\_COMPAREMODE\_DELAY\_CMP1
    - LL\_HRTIM\_COMPAREMODE\_DELAY\_CMP3
- Return values**
- **None**
- Notes**
- In auto-delayed mode the compare match occurs independently from the timer counter value.
- Reference Manual to LL API cross reference:**
- TIMxCR DELCMP2 LL\_HRTIM\_TIM\_SetCompareMode
  - TIMxCR DELCMP4 LL\_HRTIM\_TIM\_SetCompareMode

### LL\_HRTIM\_TIM\_GetCompareMode

|                                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>                               | <code>_STATIC_INLINE uint32_t LL_HRTIM_TIM_GetCompareMode(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t CompareUnit)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Function description</b>                        | Get the functioning mode of the compare unit.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Parameters</b>                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>CompareUnit:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_COMPAREUNIT_2</li> <li>– LL_HRTIM_COMPAREUNIT_4</li> </ul> </li> </ul> |
| <b>Return values</b>                               | <ul style="list-style-type: none"> <li>• <b>Mode:</b> Returned value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_COMPAREMODE_REGULAR</li> <li>– LL_HRTIM_COMPAREMODE_DELAY_NOTIMEOUT</li> <li>– LL_HRTIM_COMPAREMODE_DELAY_CMP1</li> <li>– LL_HRTIM_COMPAREMODE_DELAY_CMP3</li> </ul> </li> </ul>                                                                                                                                                                                                                                   |
| <b>Reference Manual to LL API cross reference:</b> | <ul style="list-style-type: none"> <li>• TIMxCR DELCMP2 LL_HRTIM_TIM_SetCompareMode</li> <li>• TIMxCR DELCMP4 LL_HRTIM_TIM_SetCompareMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                    |

### LL\_HRTIM\_TIM\_SetCounter

|                             |                                                                                                                                                                               |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function name</b>        | <code>_STATIC_INLINE void LL_HRTIM_TIM_SetCounter(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Counter)</code>                                                            |
| <b>Function description</b> | Set the timer counter value.                                                                                                                                                  |
| <b>Parameters</b>           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:</li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul>                                                                                                                                                                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Counter:</b> Value between 0 and 0xFFFF</li> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function can only be called when the timer is stopped.</li> <li>• For HR clock prescaling ratio below 32 (CKPSC[2:0] &lt; 5), the least significant bits of the counter are not significant. They cannot be written and return 0 when read.</li> <li>• The timer behavior is not guaranteed if the counter value is set above the period.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCNTR MCNT LL_HRTIM_TIM_SetCounter</li> <li>• CNTxR CNTx LL_HRTIM_TIM_SetCounter</li> </ul>                                                                                                                                                                                                                                                               |

### LL\_HRTIM\_TIM\_GetCounter

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetCounter(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                                 |
| Function description                              | Get actual timer counter value.                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Counter:</b> Value between 0 and 0xFFFF</li> </ul>                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCNTR MCNT LL_HRTIM_TIM_GetCounter</li> <li>• CNTxR CNTx LL_HRTIM_TIM_GetCounter</li> </ul>                                                                                                                                                                                                                                                                           |

### LL\_HRTIM\_TIM\_SetPeriod

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_TIM_SetPeriod(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Period)</code>                                                                                                                                                                                                                                                                                     |
| Function description | Set the timer period value.                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

- **Period:** Value between 0 and 0xFFFF
  - **None**
- Return values
- Reference Manual to LL API cross reference:
- MPER MPER LL\_HRTIM\_TIM\_SetPeriod
  - PERxR PERx LL\_HRTIM\_TIM\_SetPeriod

### **LL\_HRTIM\_TIM\_GetPeriod**

- |                                             |                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_TIM_GetPeriod(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                                     |
| Function description                        | Get actual timer period value.                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Period:</b> Value between 0 and 0xFFFF</li> </ul>                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MPER MPER LL_HRTIM_TIM_GetPeriod</li> <li>• PERxR PERx LL_HRTIM_TIM_GetPeriod</li> </ul>                                                                                                                                                                                                                                                                              |

### **LL\_HRTIM\_TIM\_SetRepetition**

- |                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_HRTIM_TIM_SetRepetition(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Repetition)</code></b>                                                                                                                                                                                                                                                                                                                                       |
| Function description                        | Set the timer repetition period value.                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Repetition:</b> Value between 0 and 0xFF</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MREP MREP LL_HRTIM_TIM_SetRepetition</li> <li>• REPxR REPx LL_HRTIM_TIM_SetRepetition</li> </ul>                                                                                                                                                                                                                                                                                                                             |

**LL\_HRTIM\_TIM\_GetRepetition**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_TIM_GetRepetition(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                       |
| Function description                              | Get actual timer repetition period value.                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Repetition:</b> Value between 0 and 0xFF</li> </ul>                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MREP MREP LL_HRTIM_TIM_GetRepetition</li> <li>• REPxR REPx LL_HRTIM_TIM_GetRepetition</li> </ul>                                                                                                                                                                                                                                                            |

**LL\_HRTIM\_TIM\_SetCompare1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_TIM_SetCompare1(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t CompareValue)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Set the compare value of the compare unit 1.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>CompareValue:</b> Compare value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,...</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCMP1R MCMP1 LL_HRTIM_TIM_SetCompare1</li> <li>• CMP1xR CMP1x LL_HRTIM_TIM_SetCompare1</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

**LL\_HRTIM\_TIM\_GetCompare1**

|                      |                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_TIM_GetCompare1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                         |
| Function description | Get actual compare value of the compare unit 1.                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul>                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>CompareValue:</b> Compare value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,...</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCMP1R MCMP1 LL_HRTIM_TIM_GetCompare1</li> <li>• CMP1xR CMP1x LL_HRTIM_TIM_GetCompare1</li> </ul>                                                                                         |

### LL\_HRTIM\_TIM\_SetCompare2

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_SetCompare2(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t CompareValue)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                              | Set the compare value of the compare unit 2.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>CompareValue:</b> Compare value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,...</li> </ul> |
| Return values                                     | • <b>None</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCMP2R MCMP2 LL_HRTIM_TIM_SetCompare2</li> <li>• CMP2xR CMP2x LL_HRTIM_TIM_SetCompare2</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

### LL\_HRTIM\_TIM\_GetCompare2

|                      |                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetCompare2(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                               |
| Function description | Get actual compare value of the compare unit 2.                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>CompareValue:</b> Compare value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0,</li> </ul>                                                                                                                                                                                                                      |

0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,...

Reference Manual to  
LL API cross  
reference:

- MCMP2R MCMP2 LL\_HRTIM\_TIM\_SetCompare3
- CMP2xR CMP2x LL\_HRTIM\_TIM\_SetCompare3
- 

### **LL\_HRTIM\_TIM\_SetCompare3**

Function name

**`__STATIC_INLINE void LL_HRTIM_TIM_SetCompare3(  
 HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t  
 CompareValue)`**

Function description

Set the compare value of the compare unit 3.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E
- **CompareValue:** Compare value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,...

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- MCMP3R MCMP3 LL\_HRTIM\_TIM\_SetCompare3
- CMP3xR CMP3x LL\_HRTIM\_TIM\_SetCompare3

### **LL\_HRTIM\_TIM\_GetCompare3**

Function name

**`__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetCompare3(  
 HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Get actual compare value of the compare unit 3.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **CompareValue:** Compare value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,...

Reference Manual to  
LL API cross  
reference:

- MCMP3R MCMP3 LL\_HRTIM\_TIM\_GetCompare3
- CMP3xR CMP3x LL\_HRTIM\_TIM\_GetCompare3

**LL\_HRTIM\_TIM\_SetCompare4**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_SetCompare4(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t CompareValue)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                              | Set the compare value of the compare unit 4.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>CompareValue:</b> Compare value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,...</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCMP4R MCMP4 LL_HRTIM_TIM_SetCompare4</li> <li>• CMP4xR CMP4x LL_HRTIM_TIM_SetCompare4</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

**LL\_HRTIM\_TIM\_GetCompare4**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetCompare4(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                               |
| Function description                              | Get actual compare value of the compare unit 4.                                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>CompareValue:</b> Compare value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,...</li> </ul>                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MCMP4R MCMP4 LL_HRTIM_TIM_GetCompare4</li> <li>• CMP4xR CMP4x LL_HRTIM_TIM_GetCompare4</li> </ul>                                                                                                                                                                                                                                                           |

**LL\_HRTIM\_TIM\_SetResetTrig**

|                      |                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_TIM_SetResetTrig(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t ResetTrig)</code> |
| Function description | Set the reset trigger of a timer counter.                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                       |

- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E
- **ResetTrig:** This parameter can be a combination of the following values:
  - LL\_HRTIM\_RESETTRIG\_NONE
  - LL\_HRTIM\_RESETTRIG\_UPDATE
  - LL\_HRTIM\_RESETTRIG\_CMP2
  - LL\_HRTIM\_RESETTRIG\_CMP4
  - LL\_HRTIM\_RESETTRIG\_MASTER\_PER
  - LL\_HRTIM\_RESETTRIG\_MASTER\_CMP1
  - LL\_HRTIM\_RESETTRIG\_MASTER\_CMP2
  - LL\_HRTIM\_RESETTRIG\_MASTER\_CMP3
  - LL\_HRTIM\_RESETTRIG\_MASTER\_CMP4
  - LL\_HRTIM\_RESETTRIG\_EEV\_1
  - LL\_HRTIM\_RESETTRIG\_EEV\_2
  - LL\_HRTIM\_RESETTRIG\_EEV\_3
  - LL\_HRTIM\_RESETTRIG\_EEV\_4
  - LL\_HRTIM\_RESETTRIG\_EEV\_5
  - LL\_HRTIM\_RESETTRIG\_EEV\_6
  - LL\_HRTIM\_RESETTRIG\_EEV\_7
  - LL\_HRTIM\_RESETTRIG\_EEV\_8
  - LL\_HRTIM\_RESETTRIG\_EEV\_9
  - LL\_HRTIM\_RESETTRIG\_EEV\_10
  - LL\_HRTIM\_RESETTRIG\_OTHER1\_CMP1
  - LL\_HRTIM\_RESETTRIG\_OTHER1\_CMP2
  - LL\_HRTIM\_RESETTRIG\_OTHER1\_CMP4
  - LL\_HRTIM\_RESETTRIG\_OTHER2\_CMP1
  - LL\_HRTIM\_RESETTRIG\_OTHER2\_CMP2
  - LL\_HRTIM\_RESETTRIG\_OTHER2\_CMP4
  - LL\_HRTIM\_RESETTRIG\_OTHER3\_CMP1
  - LL\_HRTIM\_RESETTRIG\_OTHER3\_CMP2
  - LL\_HRTIM\_RESETTRIG\_OTHER3\_CMP4
  - LL\_HRTIM\_RESETTRIG\_OTHER4\_CMP1
  - LL\_HRTIM\_RESETTRIG\_OTHER4\_CMP2
  - LL\_HRTIM\_RESETTRIG\_OTHER4\_CMP4

**Return values**

- **None**

**Notes**

- The reset of the timer counter can be triggered by up to 30 events that can be selected among the following sources: The timing unit: Compare 2, Compare 4 and Update (3 events). The master timer: Reset and Compare 1..4 (5 events). The external events EXTEVNT1..10 (10 events). All other timing units (e.g. Timer B..E for timer A): Compare 1, 2 and 4 (12 events).

**Reference Manual to  
LL API cross  
reference:**

- RSTxR UPDT LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR CMP2 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR CMP4 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR MSTPER LL\_HRTIM\_TIM\_SetResetTrig

- RSTxR MSTCMP1 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR MSTCMP2 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR MSTCMP3 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR MSTCMP4 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR EXTEVNT1 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR EXTEVNT2 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR EXTEVNT3 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR EXTEVNT4 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR EXTEVNT5 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR EXTEVNT6 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR EXTEVNT7 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR EXTEVNT8 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR EXTEVNT9 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR EXTEVNT10 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMBCMP1 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMBCMP2 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMBCMP4 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMCCMP1 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMCCMP2 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMCCMP4 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMDCMP1 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMDCMP2 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMDCMP4 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMECMP1 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMECMP2 LL\_HRTIM\_TIM\_SetResetTrig
- RSTxR TIMECMP4 LL\_HRTIM\_TIM\_SetResetTrig

### **LL\_HRTIM\_TIM\_GetResetTrig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>STATIC_INLINE uint32_t LL_HRTIM_TIM_GetResetTrig<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Get actual reset trigger of a timer counter.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul>                                                                                                                                                                                                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>ResetTrig:</b> Returned value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_RESETTRIG_NONE</li> <li>- LL_HRTIM_RESETTRIG_UPDATE</li> <li>- LL_HRTIM_RESETTRIG_CMP2</li> <li>- LL_HRTIM_RESETTRIG_CMP4</li> <li>- LL_HRTIM_RESETTRIG_MASTER_PER</li> <li>- LL_HRTIM_RESETTRIG_MASTER_CMP1</li> <li>- LL_HRTIM_RESETTRIG_MASTER_CMP2</li> <li>- LL_HRTIM_RESETTRIG_MASTER_CMP3</li> <li>- LL_HRTIM_RESETTRIG_MASTER_CMP4</li> <li>- LL_HRTIM_RESETTRIG_EEV_1</li> </ul> </li> </ul> |

- LL\_HRTIM\_RESETTRIG\_EEV\_2
- LL\_HRTIM\_RESETTRIG\_EEV\_3
- LL\_HRTIM\_RESETTRIG\_EEV\_4
- LL\_HRTIM\_RESETTRIG\_EEV\_5
- LL\_HRTIM\_RESETTRIG\_EEV\_6
- LL\_HRTIM\_RESETTRIG\_EEV\_7
- LL\_HRTIM\_RESETTRIG\_EEV\_8
- LL\_HRTIM\_RESETTRIG\_EEV\_9
- LL\_HRTIM\_RESETTRIG\_EEV\_10
- LL\_HRTIM\_RESETTRIG\_OTHER1\_CMP1
- LL\_HRTIM\_RESETTRIG\_OTHER1\_CMP2
- LL\_HRTIM\_RESETTRIG\_OTHER1\_CMP4
- LL\_HRTIM\_RESETTRIG\_OTHER2\_CMP1
- LL\_HRTIM\_RESETTRIG\_OTHER2\_CMP2
- LL\_HRTIM\_RESETTRIG\_OTHER2\_CMP4
- LL\_HRTIM\_RESETTRIG\_OTHER3\_CMP1
- LL\_HRTIM\_RESETTRIG\_OTHER3\_CMP2
- LL\_HRTIM\_RESETTRIG\_OTHER3\_CMP4
- LL\_HRTIM\_RESETTRIG\_OTHER4\_CMP1
- LL\_HRTIM\_RESETTRIG\_OTHER4\_CMP2
- LL\_HRTIM\_RESETTRIG\_OTHER4\_CMP4

Reference Manual to  
LL API cross  
reference:

- RSTxR UPDT LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR CMP2 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR CMP4 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR MSTPER LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR MSTCMP1 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR MSTCMP2 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR MSTCMP3 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR MSTCMP4 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR EXTEVNT1 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR EXTEVNT2 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR EXTEVNT3 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR EXTEVNT4 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR EXTEVNT5 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR EXTEVNT6 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR EXTEVNT7 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR EXTEVNT8 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR EXTEVNT9 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR EXTEVNT10 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMBCMP1 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMBCMP2 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMBCMP4 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMCCMP1 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMCCMP2 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMCCMP4 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMDCMP1 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMDCMP2 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMDCMP4 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMECMP1 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMECMP2 LL\_HRTIM\_TIM\_GetResetTrig
- RSTxR TIMECMP4 LL\_HRTIM\_TIM\_GetResetTrig

**LL\_HRTIM\_TIM\_GetCapture1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetCapture1<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                         |
| Function description                              | Get captured value for capture unit 1.                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Captured:</b> value</li> </ul>                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CPT1xR CPT1x LL_HRTIM_TIM_GetCapture1</li> </ul>                                                                                                                                                                                                                                                                           |

**LL\_HRTIM\_TIM\_GetCapture2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetCapture2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                         |
| Function description                              | Get captured value for capture unit 2.                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Captured:</b> value</li> </ul>                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CPT2xR CPT2x LL_HRTIM_TIM_GetCapture2</li> </ul>                                                                                                                                                                                                                                                                           |

**LL\_HRTIM\_TIM\_SetCaptureTrig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_TIM_SetCaptureTrig<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t<br/>CaptureUnit, uint32_t CaptureTrig)</code>                                                                                                                                                                                                                                                                                            |
| Function description | Set the trigger of a capture unit for a given timer.                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>CaptureUnit:</b> This parameter can be one of the following values:</li> </ul> |

- LL\_HRTIM\_CAPTUREUNIT\_1
- LL\_HRTIM\_CAPTUREUNIT\_2
- **CaptureTrig:** This parameter can be a combination of the following values:
  - LL\_HRTIM\_CAPTURETRIG\_NONE
  - LL\_HRTIM\_CAPTURETRIG\_UPDATE
  - LL\_HRTIM\_CAPTURETRIG\_EEV\_1
  - LL\_HRTIM\_CAPTURETRIG\_EEV\_2
  - LL\_HRTIM\_CAPTURETRIG\_EEV\_3
  - LL\_HRTIM\_CAPTURETRIG\_EEV\_4
  - LL\_HRTIM\_CAPTURETRIG\_EEV\_5
  - LL\_HRTIM\_CAPTURETRIG\_EEV\_6
  - LL\_HRTIM\_CAPTURETRIG\_EEV\_7
  - LL\_HRTIM\_CAPTURETRIG\_EEV\_8
  - LL\_HRTIM\_CAPTURETRIG\_EEV\_9
  - LL\_HRTIM\_CAPTURETRIG\_EEV\_10
  - LL\_HRTIM\_CAPTURETRIG\_TA1\_SET
  - LL\_HRTIM\_CAPTURETRIG\_TA1\_RESET
  - LL\_HRTIM\_CAPTURETRIG\_TIMA\_CMP1
  - LL\_HRTIM\_CAPTURETRIG\_TIMA\_CMP2
  - LL\_HRTIM\_CAPTURETRIG\_TB1\_SET
  - LL\_HRTIM\_CAPTURETRIG\_TB1\_RESET
  - LL\_HRTIM\_CAPTURETRIG\_TIMB\_CMP1
  - LL\_HRTIM\_CAPTURETRIG\_TIMB\_CMP2
  - LL\_HRTIM\_CAPTURETRIG\_TC1\_SET
  - LL\_HRTIM\_CAPTURETRIG\_TC1\_RESET
  - LL\_HRTIM\_CAPTURETRIG\_TIMC\_CMP1
  - LL\_HRTIM\_CAPTURETRIG\_TIMC\_CMP2
  - LL\_HRTIM\_CAPTURETRIG\_TD1\_SET
  - LL\_HRTIM\_CAPTURETRIG\_TD1\_RESET
  - LL\_HRTIM\_CAPTURETRIG\_TIMD\_CMP1
  - LL\_HRTIM\_CAPTURETRIG\_TIMD\_CMP2
  - LL\_HRTIM\_CAPTURETRIG\_TE1\_SET
  - LL\_HRTIM\_CAPTURETRIG\_TE1\_RESET
  - LL\_HRTIM\_CAPTURETRIG\_TIME\_CMP1
  - LL\_HRTIM\_CAPTURETRIG\_TIME\_CMP2

**Return values**

Reference Manual to  
LL API cross  
reference:

- **None**
- CPT1xCR SWCPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR UPDCPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR EXEV1CPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR EXEV2CPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR EXEV3CPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR EXEV4CPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR EXEV5CPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR EXEV6CPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR EXEV7CPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR EXEV8CPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR EXEV9CPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR EXEV10CPT LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TA1SET LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TA1RST LL\_HRTIM\_TIM\_SetCaptureTrig

- CPT1xCR TACMP1 LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TACMP2 LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TB1SET LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TB1RST LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TBCMP1 LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TBCMP2 LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TC1SET LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TC1RST LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TCCMP1 LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TCCMP2 LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TD1SET LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TD1RST LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TDCMP1 LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TDCMP2 LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TE1SET LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TE1RST LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TECMP1 LL\_HRTIM\_TIM\_SetCaptureTrig
- CPT1xCR TECMP2 LL\_HRTIM\_TIM\_SetCaptureTrig

### **LL\_HRTIM\_TIM\_GetCaptureTrig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetCaptureTrig(<br/>  HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t<br/>  CaptureUnit)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description | Get actual trigger of a capture unit for a given timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>CaptureUnit:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_CAPTUREUNIT_1</li> <li>- LL_HRTIM_CAPTUREUNIT_2</li> </ul> </li> </ul>                                                                                                                             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>CaptureTrig:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_CAPTURETRIG_NONE</li> <li>- LL_HRTIM_CAPTURETRIG_UPDATE</li> <li>- LL_HRTIM_CAPTURETRIG_EEV_1</li> <li>- LL_HRTIM_CAPTURETRIG_EEV_2</li> <li>- LL_HRTIM_CAPTURETRIG_EEV_3</li> <li>- LL_HRTIM_CAPTURETRIG_EEV_4</li> <li>- LL_HRTIM_CAPTURETRIG_EEV_5</li> <li>- LL_HRTIM_CAPTURETRIG_EEV_6</li> <li>- LL_HRTIM_CAPTURETRIG_EEV_7</li> <li>- LL_HRTIM_CAPTURETRIG_EEV_8</li> <li>- LL_HRTIM_CAPTURETRIG_EEV_9</li> <li>- LL_HRTIM_CAPTURETRIG_EEV_10</li> <li>- LL_HRTIM_CAPTURETRIG_TA1_SET</li> </ul> </li> </ul> |

- LL\_HRTIM\_CAPTURETRIG\_TA1\_RESET
- LL\_HRTIM\_CAPTURETRIG\_TIMA\_CMP1
- LL\_HRTIM\_CAPTURETRIG\_TIMA\_CMP2
- LL\_HRTIM\_CAPTURETRIG\_TB1\_SET
- LL\_HRTIM\_CAPTURETRIG\_TB1\_RESET
- LL\_HRTIM\_CAPTURETRIG\_TIMB\_CMP1
- LL\_HRTIM\_CAPTURETRIG\_TIMB\_CMP2
- LL\_HRTIM\_CAPTURETRIG\_TC1\_SET
- LL\_HRTIM\_CAPTURETRIG\_TC1\_RESET
- LL\_HRTIM\_CAPTURETRIG\_TIMC\_CMP1
- LL\_HRTIM\_CAPTURETRIG\_TIMC\_CMP2
- LL\_HRTIM\_CAPTURETRIG\_TD1\_SET
- LL\_HRTIM\_CAPTURETRIG\_TD1\_RESET
- LL\_HRTIM\_CAPTURETRIG\_TIMD\_CMP1
- LL\_HRTIM\_CAPTURETRIG\_TIMD\_CMP2
- LL\_HRTIM\_CAPTURETRIG\_TE1\_SET
- LL\_HRTIM\_CAPTURETRIG\_TE1\_RESET
- LL\_HRTIM\_CAPTURETRIG\_TIME\_CMP1
- LL\_HRTIM\_CAPTURETRIG\_TIME\_CMP2

Reference Manual to  
LL API cross  
reference:

- CPT1xCR SWCPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR UPDCPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR EXEV1CPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR EXEV2CPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR EXEV3CPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR EXEV4CPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR EXEV5CPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR EXEV6CPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR EXEV7CPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR EXEV8CPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR EXEV9CPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR EXEV10CPT LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TA1SET LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TA1RST LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TACMP1 LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TACMP2 LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TB1SET LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TB1RST LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TBCMP1 LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TBCMP2 LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TC1SET LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TC1RST LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TCCMP1 LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TCCMP2 LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TD1SET LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TD1RST LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TDCMP1 LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TDCMP2 LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TE1SET LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TE1RST LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TECMP1 LL\_HRTIM\_TIM\_GetCaptureTrig
- CPT1xCR TECMP2 LL\_HRTIM\_TIM\_GetCaptureTrig

**LL\_HRTIM\_TIM\_EnableDeadTime**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_EnableDeadTime<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                          |
| Function description                              | Enable deadtime insertion for a given timer.                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR DTEN LL_HRTIM_TIM_EnableDeadTime</li> </ul>                                                                                                                                                                                                                                                                          |

**LL\_HRTIM\_TIM\_DisableDeadTime**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_DisableDeadTime<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                         |
| Function description                              | Disable deadtime insertion for a given timer.                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR DTEN LL_HRTIM_TIM_DisableDeadTime</li> </ul>                                                                                                                                                                                                                                                                         |

**LL\_HRTIM\_TIM\_IsEnabledDeadTime**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_TIM_IsEnabledDeadTime (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                               |
| Function description | Indicate whether deadtime insertion is enabled for a given timer.                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of DTEN bit in HRTIM_OUTxR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                     |

- Reference Manual to LL API cross reference:
- OUTxR DTEN LL\_HRTIM\_TIM\_IsEnabledDeadTime

### **LL\_HRTIM\_TIM\_SetDLYPRTMode**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_HRTIM_TIM_SetDLYPRTMode(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t<br/>    DLYPRTMode)</code>                                                                                                                                                                                                                                                                                             |
| Function description                        | Set the delayed protection (DLYPRT) mode.                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li><b>DLYPRTMode:</b> Delayed protection (DLYPRT) mode</li> </ul> |
| Notes                                       | <ul style="list-style-type: none"> <li>This function must be called prior enabling the delayed protection</li> <li>Balanced Idle mode is only available in push-pull mode</li> </ul>                                                                                                                                                                                                                                                   |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>OUTxR DLYPRTEN LL_HRTIM_TIM_SetDLYPRTMode</li> <li>OUTxR DLYPRT LL_HRTIM_TIM_SetDLYPRTMode</li> </ul>                                                                                                                                                                                                                                                                                           |

### **LL\_HRTIM\_TIM\_GetDLYPRTMode**

|                                             |                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetDLYPRTMode(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                         |
| Function description                        | Get the delayed protection (DLYPRT) mode.                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>DLYPRTMode:</b> Delayed protection (DLYPRT) mode</li> </ul>                                                                                                                                                                                                                                                                     |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>OUTxR DLYPRTEN LL_HRTIM_TIM_GetDLYPRTMode</li> <li>OUTxR DLYPRT LL_HRTIM_TIM_GetDLYPRTMode</li> </ul>                                                                                                                                                                                                                              |

### **LL\_HRTIM\_TIM\_EnableDLYPRT**

|                      |                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_TIM_EnableDLYPRT(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                              |
| Function description | Enable delayed protection (DLYPRT) for a given timer.                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values:</li> </ul> |

---

|                                                   |                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called once the concerned timer is enabled</li> </ul>                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR DLYPRTEN LL_HRTIM_TIM_EnableDLYPRT</li> </ul>                                                                                         |

### LL\_HRTIM\_TIM\_DisableDLYPRT

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_DisableDLYPRT<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                           |
| Function description                              | Disable delayed protection (DLYPRT) for a given timer.                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called once the concerned timer is enabled</li> </ul>                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR DLYPRTEN LL_HRTIM_TIM_DisableDLYPRT</li> </ul>                                                                                                                                                                                                                                                                       |

### LL\_HRTIM\_TIM\_IsEnabledDLYPRT

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_IsEnabledDLYPRT<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                     |
| Function description | Indicate whether delayed protection (DLYPRT) is enabled for a given timer.                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of DLYPRTEN bit in HRTIM_OUTxR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                 |

- Reference Manual to  
LL API cross  
reference:
- OUTxR DLYPRTEN LL\_HRTIM\_TIM\_IsEnabledDLYPRT

### **LL\_HRTIM\_TIM\_EnableFault**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_EnableFault<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Faults)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description                              | Enable the fault channel(s) for a given timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Faults:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_FAULT_1</li> <li>– LL_HRTIM_FAULT_2</li> <li>– LL_HRTIM_FAULT_3</li> <li>– LL_HRTIM_FAULT_4</li> <li>– LL_HRTIM_FAULT_5</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• FLTxA R FLT1EN LL_HRTIM_TIM_EnableFault</li> <li>• FLTxB R FLT2EN LL_HRTIM_TIM_EnableFault</li> <li>• FLTxC R FLT3EN LL_HRTIM_TIM_EnableFault</li> <li>• FLTxD R FLT4EN LL_HRTIM_TIM_EnableFault</li> <li>• FLTxE R FLT5EN LL_HRTIM_TIM_EnableFault</li> </ul>                                                                                                                                                                                                                                                                                                                                          |

### **LL\_HRTIM\_TIM\_DisableFault**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_TIM_DisableFault<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Faults)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Disable the fault channel(s) for a given timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Faults:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_FAULT_1</li> <li>– LL_HRTIM_FAULT_2</li> <li>– LL_HRTIM_FAULT_3</li> <li>– LL_HRTIM_FAULT_4</li> <li>– LL_HRTIM_FAULT_5</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>FLTxA R FLT1EN LL_HRTIM_TIM_DisableFault</li> <li>FLTxA R FLT2EN LL_HRTIM_TIM_DisableFault</li> <li>FLTxA R FLT3EN LL_HRTIM_TIM_DisableFault</li> <li>FLTxA R FLT4EN LL_HRTIM_TIM_DisableFault</li> <li>FLTxA R FLT5EN LL_HRTIM_TIM_DisableFault</li> </ul> |

### LL\_HRTIM\_TIM\_IsEnabledFault

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_IsEnabledFault(<br/>HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Fault)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description                              | Indicate whether the fault channel is enabled for a given timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li><b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_FAULT_1</li> <li>– LL_HRTIM_FAULT_2</li> <li>– LL_HRTIM_FAULT_3</li> <li>– LL_HRTIM_FAULT_4</li> <li>– LL_HRTIM_FAULT_5</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of FLTxAEN bit in HRTIM_FLTxR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>FLTxA R FLT1EN LL_HRTIM_TIM_IsEnabledFault</li> <li>FLTxA R FLT2EN LL_HRTIM_TIM_IsEnabledFault</li> <li>FLTxA R FLT3EN LL_HRTIM_TIM_IsEnabledFault</li> <li>FLTxA R FLT4EN LL_HRTIM_TIM_IsEnabledFault</li> <li>FLTxA R FLT5EN LL_HRTIM_TIM_IsEnabledFault</li> </ul>                                                                                                                                                                                                                                                                                                                    |

### LL\_HRTIM\_TIM\_LockFault

|                      |                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_TIM_LockFault(<br/>HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                           |
| Function description | Lock the fault conditioning set-up for a given timer.                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                   |
| Notes                | <ul style="list-style-type: none"> <li>Timer fault-related set-up is frozen until the next HRTIM or system reset</li> </ul>                                                                                                                                                                                                                                     |

Reference Manual to  
LL API cross  
reference:

- `FLTxD FLTLCK LL_HRTIM_TIM_LockFault`

### `LL_HRTIM_TIM_SetBurstModeOption`

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_TIM_SetBurstModeOption (HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t BurstsModeOption)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Define how the timer behaves during a burst mode operation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_HRTIM_TIMER_MASTER</code></li> <li>– <code>LL_HRTIM_TIMER_A</code></li> <li>– <code>LL_HRTIM_TIMER_B</code></li> <li>– <code>LL_HRTIM_TIMER_C</code></li> <li>– <code>LL_HRTIM_TIMER_D</code></li> <li>– <code>LL_HRTIM_TIMER_E</code></li> </ul> </li> <li>• <b>BurstsModeOption:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_HRTIM_BURSTMODE_MAINTAINCLOCK</code></li> <li>– <code>LL_HRTIM_BURSTMODE_RESETCOUNTER</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the burst mode is enabled</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>BMCR MTBM LL_HRTIM_TIM_SetBurstModeOption</code></li> <li>• <code>BMCR TABM LL_HRTIM_TIM_SetBurstModeOption</code></li> <li>• <code>BMCR TBBM LL_HRTIM_TIM_SetBurstModeOption</code></li> <li>• <code>BMCR TCBM LL_HRTIM_TIM_SetBurstModeOption</code></li> <li>• <code>BMCR TDBM LL_HRTIM_TIM_SetBurstModeOption</code></li> <li>• <code>BMCR TEBM LL_HRTIM_TIM_SetBurstModeOption</code></li> </ul>                                                                                                                                                                                                                                                                                 |

### `LL_HRTIM_TIM_GetBurstModeOption`

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetBurstModeOption (HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Retrieve how the timer behaves during a burst mode operation.                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_HRTIM_TIMER_MASTER</code></li> <li>– <code>LL_HRTIM_TIMER_A</code></li> <li>– <code>LL_HRTIM_TIMER_B</code></li> <li>– <code>LL_HRTIM_TIMER_C</code></li> <li>– <code>LL_HRTIM_TIMER_D</code></li> <li>– <code>LL_HRTIM_TIMER_E</code></li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>BurstsMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_HRTIM_BURSTMODE_MAINTAINCLOCK</code></li> </ul> </li> </ul>                                                                                                                                                                                                                                                       |

Reference Manual to  
LL API cross  
reference:

- LL\_HRTIM\_BURSTMODE\_RESETCOUNTER

- BMCR MCR LL\_HRTIM\_TIM\_GetBurstModeOption
- BMCR TABM LL\_HRTIM\_TIM\_GetBurstModeOption
- BMCR TBBM LL\_HRTIM\_TIM\_GetBurstModeOption
- BMCR TCBM LL\_HRTIM\_TIM\_GetBurstModeOption
- BMCR TDBM LL\_HRTIM\_TIM\_GetBurstModeOption
- BMCR TEBM LL\_HRTIM\_TIM\_GetBurstModeOption

### **LL\_HRTIM\_TIM\_ConfigBurstDMA**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_TIM_ConfigBurstDMA(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Registers)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description                              | Program which registers are to be written by Burst DMA transfers.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Registers:</b> Registers to be updated by the DMA request</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDMUPDR MTBM LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDMUPDR MICR LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDMUPDR MDIER LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDMUPDR MCNT LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDMUPDR MPER LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDMUPDR MREP LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDMUPDR MCMP1 LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDMUPDR MCMP2 LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDMUPDR MCMP3 LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDMUPDR MCMP4 LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxCR LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxICR LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxDIER LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxCNT LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxPER LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxREP LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxCMP1 LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxCMP2 LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxCMP3 LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxCMP4 LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxDTR LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxSET1R LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxRST1R LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxSET2R LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxRST2R LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIAEEFR1 LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxEEFR2 LL_HRTIM_TIM_ConfigBurstDMA</li> <li>• BDTxUPDR TIMxRSTR LL_HRTIM_TIM_ConfigBurstDMA</li> </ul> |

- BDTxUPDR TIMxOUTR LL\_HRTIM\_TIM\_ConfigBurstDMA
- BDTxUPDR TIMxLTCH LL\_HRTIM\_TIM\_ConfigBurstDMA

### **LL\_HRTIM\_TIM\_GetCurrentPushPullStatus**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t<br/>LL_HRTIM_TIM_GetCurrentPushPullStatus (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                  |
| Function description                              | Indicate on which output the signal is currently applied.                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>CPPSTAT:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_CPPSTAT_OUTPUT1</li> <li>– LL_HRTIM_CPPSTAT_OUTPUT2</li> </ul> </li> </ul>                                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• Only significant when the timer operates in push-pull mode.</li> </ul>                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxISR CPPSTAT<br/><code>LL_HRTIM_TIM_GetCurrentPushPullStatus</code></li> </ul>                                                                                                                                                                                                                                          |

### **LL\_HRTIM\_TIM\_GetIdlePushPullStatus**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t<br/>LL_HRTIM_TIM_GetIdlePushPullStatus (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                     |
| Function description                              | Indicate on which output the signal was applied, in push-pull mode, balanced fault mode or delayed idle mode, when the protection was triggered.                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>IPPSTAT:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_IPPSTAT_OUTPUT1</li> <li>– LL_HRTIM_IPPSTAT_OUTPUT2</li> </ul> </li> </ul>                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxISR IPPSTAT <code>LL_HRTIM_TIM_GetIdlePushPullStatus</code></li> </ul>                                                                                                                                                                                                                                                 |

**LL\_HRTIM\_TIM\_SetEventFilter**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_HRTIM_TIM_SetEventFilter(<br/>    (HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Event,<br/>    uint32_t Filter)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description                              | Set the event filter for a given timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EVENT_1</li> <li>- LL_HRTIM_EVENT_2</li> <li>- LL_HRTIM_EVENT_3</li> <li>- LL_HRTIM_EVENT_4</li> <li>- LL_HRTIM_EVENT_5</li> <li>- LL_HRTIM_EVENT_6</li> <li>- LL_HRTIM_EVENT_7</li> <li>- LL_HRTIM_EVENT_8</li> <li>- LL_HRTIM_EVENT_9</li> <li>- LL_HRTIM_EVENT_10</li> </ul> </li> <li>• <b>Filter:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EEFLTR_NONE</li> <li>- LL_HRTIM_EEFLTR_BLANKINGCMP1</li> <li>- LL_HRTIM_EEFLTR_BLANKINGCMP2</li> <li>- LL_HRTIM_EEFLTR_BLANKINGCMP3</li> <li>- LL_HRTIM_EEFLTR_BLANKINGFLTR1</li> <li>- LL_HRTIM_EEFLTR_BLANKINGFLTR2</li> <li>- LL_HRTIM_EEFLTR_BLANKINGFLTR3</li> <li>- LL_HRTIM_EEFLTR_BLANKINGFLTR4</li> <li>- LL_HRTIM_EEFLTR_BLANKINGFLTR5</li> <li>- LL_HRTIM_EEFLTR_BLANKINGFLTR6</li> <li>- LL_HRTIM_EEFLTR_BLANKINGFLTR7</li> <li>- LL_HRTIM_EEFLTR_BLANKINGFLTR8</li> <li>- LL_HRTIM_EEFLTR_WINDOWINGCMP2</li> <li>- LL_HRTIM_EEFLTR_WINDOWINGCMP3</li> <li>- LL_HRTIM_EEFLTR_WINDOWINGTIM</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the timer counter is enabled.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EEFxR1 EE1LTCH LL_HRTIM_TIM_SetEventFilter</li> <li>• EEFxR1 EE2LTCH LL_HRTIM_TIM_SetEventFilter</li> <li>• EEFxR1 EE3LTCH LL_HRTIM_TIM_SetEventFilter</li> <li>• EEFxR1 EE4LTCH LL_HRTIM_TIM_SetEventFilter</li> <li>• EEFxR1 EE5LTCH LL_HRTIM_TIM_SetEventFilter</li> <li>• EEFxR2 EE6LTCH LL_HRTIM_TIM_SetEventFilter</li> <li>• EEFxR2 EE7LTCH LL_HRTIM_TIM_SetEventFilter</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

- EEFxR2 EE8LTCH LL\_HRTIM\_TIM\_SetEventFilter
- EEFxR2 EE9LTCH LL\_HRTIM\_TIM\_SetEventFilter
- EEFxR2 EE10LTCH LL\_HRTIM\_TIM\_SetEventFilter

### **LL\_HRTIM\_TIM\_GetEventFilter**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_TIM_GetEventFilter(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Event)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Get actual event filter settings for a given timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Event:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_HRTIM_EVENT_1</li> <li>– LL_HRTIM_EVENT_2</li> <li>– LL_HRTIM_EVENT_3</li> <li>– LL_HRTIM_EVENT_4</li> <li>– LL_HRTIM_EVENT_5</li> <li>– LL_HRTIM_EVENT_6</li> <li>– LL_HRTIM_EVENT_7</li> <li>– LL_HRTIM_EVENT_8</li> <li>– LL_HRTIM_EVENT_9</li> <li>– LL_HRTIM_EVENT_10</li> </ul> </li> </ul>                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Filter:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_HRTIM_EEFLTR_NONE</li> <li>– LL_HRTIM_EEFLTR_BLANKINGCMP1</li> <li>– LL_HRTIM_EEFLTR_BLANKINGCMP2</li> <li>– LL_HRTIM_EEFLTR_BLANKINGCMP3</li> <li>– LL_HRTIM_EEFLTR_BLANKINGCMP4</li> <li>– LL_HRTIM_EEFLTR_BLANKINGFLTR1</li> <li>– LL_HRTIM_EEFLTR_BLANKINGFLTR2</li> <li>– LL_HRTIM_EEFLTR_BLANKINGFLTR3</li> <li>– LL_HRTIM_EEFLTR_BLANKINGFLTR4</li> <li>– LL_HRTIM_EEFLTR_BLANKINGFLTR5</li> <li>– LL_HRTIM_EEFLTR_BLANKINGFLTR6</li> <li>– LL_HRTIM_EEFLTR_BLANKINGFLTR7</li> <li>– LL_HRTIM_EEFLTR_BLANKINGFLTR8</li> <li>– LL_HRTIM_EEFLTR_WINDOWINGCMP2</li> <li>– LL_HRTIM_EEFLTR_WINDOWINGCMP3</li> <li>– LL_HRTIM_EEFLTR_WINDOWINGTIM</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EEFxR1 EE1FLTR LL_HRTIM_TIM_GetEventFilter</li> <li>• EEFxR1 EE2FLTR LL_HRTIM_TIM_GetEventFilter</li> <li>• EEFxR1 EE3FLTR LL_HRTIM_TIM_GetEventFilter</li> <li>• EEFxR1 EE4FLTR LL_HRTIM_TIM_GetEventFilter</li> <li>• EEFxR1 EE5FLTR LL_HRTIM_TIM_GetEventFilter</li> <li>• EEFxR2 EE6FLTR LL_HRTIM_TIM_GetEventFilter</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

- EEFxR2 EE7FLTR LL\_HRTIM\_TIM\_GetEventFilter
- EEFxR2 EE8FLTR LL\_HRTIM\_TIM\_GetEventFilter
- EEFxR2 EE9FLTR LL\_HRTIM\_TIM\_GetEventFilter
- EEFxR2 EE10FLTR LL\_HRTIM\_TIM\_GetEventFilter

### **LL\_HRTIM\_TIM\_SetEventLatchStatus**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_TIM_SetEventLatchStatus(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Event, uint32_t LatchStatus)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description                              | Enable or disable event latch mechanism for a given timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EVENT_1</li> <li>- LL_HRTIM_EVENT_2</li> <li>- LL_HRTIM_EVENT_3</li> <li>- LL_HRTIM_EVENT_4</li> <li>- LL_HRTIM_EVENT_5</li> <li>- LL_HRTIM_EVENT_6</li> <li>- LL_HRTIM_EVENT_7</li> <li>- LL_HRTIM_EVENT_8</li> <li>- LL_HRTIM_EVENT_9</li> <li>- LL_HRTIM_EVENT_10</li> </ul> </li> <li>• <b>LatchStatus:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EELATCH_DISABLED</li> <li>- LL_HRTIM_EELATCH_ENABLED</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the timer counter is enabled.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EEFxR1 EE1LTCH LL_HRTIM_TIM_SetEventLatchStatus</li> <li>• EEFxR1 EE2LTCH LL_HRTIM_TIM_SetEventLatchStatus</li> <li>• EEFxR1 EE3LTCH LL_HRTIM_TIM_SetEventLatchStatus</li> <li>• EEFxR1 EE4LTCH LL_HRTIM_TIM_SetEventLatchStatus</li> <li>• EEFxR1 EE5LTCH LL_HRTIM_TIM_SetEventLatchStatus</li> <li>• EEFxR2 EE6LTCH LL_HRTIM_TIM_SetEventLatchStatus</li> <li>• EEFxR2 EE7LTCH LL_HRTIM_TIM_SetEventLatchStatus</li> <li>• EEFxR2 EE8LTCH LL_HRTIM_TIM_SetEventLatchStatus</li> <li>• EEFxR2 EE9LTCH LL_HRTIM_TIM_SetEventLatchStatus</li> <li>• EEFxR2 EE10LTCH LL_HRTIM_TIM_SetEventLatchStatus</li> </ul>                                                                                                                                                                                                                                                                                                                                 |

**LL\_HRTIM\_TIM\_GetEventLatchStatus**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_TIM_GetEventLatchStatus (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer, uint32_t Event)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Get actual event latch status for a given timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_EVENT_1</li> <li>– LL_HRTIM_EVENT_2</li> <li>– LL_HRTIM_EVENT_3</li> <li>– LL_HRTIM_EVENT_4</li> <li>– LL_HRTIM_EVENT_5</li> <li>– LL_HRTIM_EVENT_6</li> <li>– LL_HRTIM_EVENT_7</li> <li>– LL_HRTIM_EVENT_8</li> <li>– LL_HRTIM_EVENT_9</li> <li>– LL_HRTIM_EVENT_10</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>LatchStatus:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_EELATCH_DISABLED</li> <li>– LL_HRTIM_EELATCH_ENABLED</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EEFxR1 EE1LTCH LL_HRTIM_TIM_GetEventLatchStatus</li> <li>• EEFxR1 EE2LTCH LL_HRTIM_TIM_GetEventLatchStatus</li> <li>• EEFxR1 EE3LTCH LL_HRTIM_TIM_GetEventLatchStatus</li> <li>• EEFxR1 EE4LTCH LL_HRTIM_TIM_GetEventLatchStatus</li> <li>• EEFxR1 EE5LTCH LL_HRTIM_TIM_GetEventLatchStatus</li> <li>• EEFxR2 EE6LTCH LL_HRTIM_TIM_GetEventLatchStatus</li> <li>• EEFxR2 EE7LTCH LL_HRTIM_TIM_GetEventLatchStatus</li> <li>• EEFxR2 EE8LTCH LL_HRTIM_TIM_GetEventLatchStatus</li> <li>• EEFxR2 EE9LTCH LL_HRTIM_TIM_GetEventLatchStatus</li> <li>• EEFxR2 EE10LTCH LL_HRTIM_TIM_GetEventLatchStatus</li> </ul>                                                                                                                            |

**LL\_HRTIM\_DT\_Config**

|                      |                                                                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_DT_Config<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t<br/>Configuration)</code>                                                                                                                                                                               |
| Function description | Configure the dead time insertion feature for a given timer.                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> </ul> </li> </ul> |

- LL\_HRTIM\_TIMER\_D
- LL\_HRTIM\_TIMER\_E
- **Configuration:** This parameter must be a combination of all the following values:
  - LL\_HRTIM\_DT\_PRESCALER\_MUL8 or ... or  
LL\_HRTIM\_DT\_PRESCALER\_DIV16
  - LL\_HRTIM\_DT\_RISING\_POSITIVE or  
LL\_HRTIM\_DT\_RISING\_NEGATIVE
  - LL\_HRTIM\_DT\_FALLING\_POSITIVE or  
LL\_HRTIM\_DT\_FALLING\_NEGATIVE

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- DTxR DTPRSC LL\_HRTIM\_DT\_Config
- DTxR SDTF LL\_HRTIM\_DT\_Config
- DTxR SDRT LL\_HRTIM\_DT\_Config

**LL\_HRTIM\_DT\_SetPrescaler**

Function name

**STATIC\_INLINE void LL\_HRTIM\_DT\_SetPrescaler  
(HRTIM\_TypeDef \* HRTIMx, uint32\_t Timer, uint32\_t Prescaler)**

Function description

Set the deadtime prescaler value.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E
- **Prescaler:** This parameter can be one of the following values:
  - LL\_HRTIM\_DT\_PRESCALER\_MUL8
  - LL\_HRTIM\_DT\_PRESCALER\_MUL4
  - LL\_HRTIM\_DT\_PRESCALER\_MUL2
  - LL\_HRTIM\_DT\_PRESCALER\_DIV1
  - LL\_HRTIM\_DT\_PRESCALER\_DIV2
  - LL\_HRTIM\_DT\_PRESCALER\_DIV4
  - LL\_HRTIM\_DT\_PRESCALER\_DIV8
  - LL\_HRTIM\_DT\_PRESCALER\_DIV16

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- DTxR DTPRSC LL\_HRTIM\_DT\_SetPrescaler

**LL\_HRTIM\_DT\_GetPrescaler**

Function name

**STATIC\_INLINE uint32\_t LL\_HRTIM\_DT\_GetPrescaler  
(HRTIM\_TypeDef \* HRTIMx, uint32\_t Timer)**

Function description

Get actual deadtime prescaler value.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                             | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul>                                                                                                                                                                                                                                                                                                          |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Prescaler:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_DT_PRESCALER_MUL8</li> <li>- LL_HRTIM_DT_PRESCALER_MUL4</li> <li>- LL_HRTIM_DT_PRESCALER_MUL2</li> <li>- LL_HRTIM_DT_PRESCALER_DIV1</li> <li>- LL_HRTIM_DT_PRESCALER_DIV2</li> <li>- LL_HRTIM_DT_PRESCALER_DIV4</li> <li>- LL_HRTIM_DT_PRESCALER_DIV8</li> <li>- LL_HRTIM_DT_PRESCALER_DIV16</li> </ul> </li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• DTxR DTPRSC LL_HRTIM_DT_GetPrescaler</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                      |

### LL\_HRTIM\_DT\_SetRisingValue

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_HRTIM_DT_SetRisingValue(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t RisingValue)</code>                                                                                                                                                                                                                                                                                                             |
| Function description                        | Set the deadtime rising value.                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>RisingValue:</b> Value between 0 and 0x1FF</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• DTxR DTR LL_HRTIM_DT_SetRisingValue</li> </ul>                                                                                                                                                                                                                                                                                                                                                |

### LL\_HRTIM\_DT\_GetRisingValue

|                      |                                                                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_DT_GetRisingValue(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                          |
| Function description | Get actual deadtime rising value.                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> </ul> </li> </ul> |

- LL\_HRTIM\_TIMER\_E
  - **RisingValue:** Value between 0 and 0x1FF
  - DTxR DTR LL\_HRTIM\_DT\_GetRisingValue
- Return values
- Reference Manual to  
LL API cross  
reference:

### LL\_HRTIM\_DT\_SetRisingSign

- |                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DT_SetRisingSign<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t<br/>RisingSign)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Set the deadtime sign on rising edge.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>RisingSign:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_DT_RISING_POSITIVE</li> <li>- LL_HRTIM_DT_RISING_NEGATIVE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | DTxR SDTR LL_HRTIM_DT_SetRisingSign                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

### LL\_HRTIM\_DT\_GetRisingSign

- |                                                   |                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_DT_GetRisingSign<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                  |
| Function description                              | Get actual deadtime sign on rising edge.                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>RisingSign:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_DT_RISING_POSITIVE</li> <li>- LL_HRTIM_DT_RISING_NEGATIVE</li> </ul> </li> </ul>                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DTxR SDTR LL_HRTIM_DT_SetRisingSign</li> </ul>                                                                                                                                                                                                                                                                                       |

**LL\_HRTIM\_DT\_SetFallingValue**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DT_SetFallingValue<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t<br/>FallingValue)</code>                                                                                                                                                                                                                                                                                         |
| Function description                              | Set the deadime falling value.                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>FallingValue:</b> Value between 0 and 0x1FF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DTxR DTF LL_HRTIM_DT_SetFallingValue</li> </ul>                                                                                                                                                                                                                                                                                                                                      |

**LL\_HRTIM\_DT\_GetFallingValue**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_DT_GetFallingValue<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                      |
| Function description                              | Get actual deadtime falling value.                                                                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>FallingValue:</b> Value between 0 and 0x1FF</li> </ul>                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DTxR DTF LL_HRTIM_DT_GetFallingValue</li> </ul>                                                                                                                                                                                                                                                                            |

**LL\_HRTIM\_DT\_SetFallingSign**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_DT_SetFallingSign<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t<br/>FallingSign)</code>                                                                                                                                                                                                                                 |
| Function description | Set the deadtime sign on falling edge.                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

- **FallingSign:** This parameter can be one of the following values:
  - LL\_HRTIM\_DT\_FALLING\_POSITIVE
  - LL\_HRTIM\_DT\_FALLING\_NEGATIVE
- **Return values**
- **Reference Manual to LL API cross reference:**
- **None**
- DTxR SDTF LL\_HRTIM\_DT\_SetFallingSign

### **LL\_HRTIM\_DT\_GetFallingSign**

- Function name      **`_STATIC_INLINE uint32_t LL_HRTIM_DT_GetFallingSign(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**
- Function description      Get actual deadtime sign on falling edge.
- Parameters
  - **HRTIMx:** High Resolution Timer instance
  - **Timer:** This parameter can be one of the following values:
    - LL\_HRTIM\_TIMER\_A
    - LL\_HRTIM\_TIMER\_B
    - LL\_HRTIM\_TIMER\_C
    - LL\_HRTIM\_TIMER\_D
    - LL\_HRTIM\_TIMER\_E
- Return values
  - **FallingSign:** This parameter can be one of the following values:
    - LL\_HRTIM\_DT\_FALLING\_POSITIVE
    - LL\_HRTIM\_DT\_FALLING\_NEGATIVE
- Reference Manual to LL API cross reference:
  - DTxR SDTF LL\_HRTIM\_DT\_GetFallingSign

### **LL\_HRTIM\_DT\_LockRising**

- Function name      **`_STATIC_INLINE void LL_HRTIM_DT_LockRising(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**
- Function description      Lock the deadtime value and sign on rising edge.
- Parameters
  - **HRTIMx:** High Resolution Timer instance
  - **Timer:** This parameter can be one of the following values:
    - LL\_HRTIM\_TIMER\_A
    - LL\_HRTIM\_TIMER\_B
    - LL\_HRTIM\_TIMER\_C
    - LL\_HRTIM\_TIMER\_D
    - LL\_HRTIM\_TIMER\_E
- Return values
  - **None**
- Reference Manual to LL API cross reference:
  - DTxR DTRLK LL\_HRTIM\_DT\_LockRising

**LL\_HRTIM\_DT\_LockRisingSign**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DT_LockRisingSign(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                |
| Function description                              | Lock the deadtime sign on rising edge.                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DTxR DTRSLK LL_HRTIM_DT_LockRisingSign</li> </ul>                                                                                                                                                                                                                                                                          |

**LL\_HRTIM\_DT\_LockFalling**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DT_LockFalling(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                   |
| Function description                              | Lock the deadtime value and sign on falling edge.                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DTxR DTFLK LL_HRTIM_DT_LockFalling</li> </ul>                                                                                                                                                                                                                                                                              |

**LL\_HRTIM\_DT\_LockFallingSign**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_DT_LockFallingSign(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                               |
| Function description | Lock the deadtime sign on falling edge.                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |

- Reference Manual to  
LL API cross  
reference:
- DTxR DTFSLK LL\_HRTIM\_DT\_LockFallingSign

### LL\_HRTIM\_CHP\_Config

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_CHP_Config<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Configuration)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Configure the chopper stage for a given timer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Configuration:</b> This parameter must be a combination of all the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_CHP_PRESCALER_DIV16 or ... or LL_HRTIM_CHP_PRESCALER_DIV256</li> <li>– LL_HRTIM_CHP_DUTYCYCLE_0 or ... or LL_HRTIM_CHP_DUTYCYCLE_875</li> <li>– LL_HRTIM_CHP_PULSEWIDTH_16 or ... or LL_HRTIM_CHP_PULSEWIDTH_256</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called if the chopper mode is already enabled for one of the timer outputs.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CHPxR CARFRQ LL_HRTIM_CHP_Config</li> <li>• CHPxR CARDTY LL_HRTIM_CHP_Config</li> <li>• CHPxR STRTPW LL_HRTIM_CHP_Config</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

### LL\_HRTIM\_CHP\_SetPrescaler

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_CHP_SetPrescaler<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t Prescaler)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Set prescaler determining the carrier frequency to be added on top of the timer output signals when chopper mode is enabled.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>Prescaler:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_CHP_PRESCALER_DIV16</li> <li>– LL_HRTIM_CHP_PRESCALER_DIV32</li> <li>– LL_HRTIM_CHP_PRESCALER_DIV48</li> </ul> </li> </ul> |

- LL\_HRTIM\_CHP\_PRESCALER\_DIV64
- LL\_HRTIM\_CHP\_PRESCALER\_DIV80
- LL\_HRTIM\_CHP\_PRESCALER\_DIV96
- LL\_HRTIM\_CHP\_PRESCALER\_DIV112
- LL\_HRTIM\_CHP\_PRESCALER\_DIV128
- LL\_HRTIM\_CHP\_PRESCALER\_DIV144
- LL\_HRTIM\_CHP\_PRESCALER\_DIV160
- LL\_HRTIM\_CHP\_PRESCALER\_DIV176
- LL\_HRTIM\_CHP\_PRESCALER\_DIV192
- LL\_HRTIM\_CHP\_PRESCALER\_DIV208
- LL\_HRTIM\_CHP\_PRESCALER\_DIV224
- LL\_HRTIM\_CHP\_PRESCALER\_DIV240
- LL\_HRTIM\_CHP\_PRESCALER\_DIV256

**Return values**

- **None**

**Notes**

- This function must not be called if the chopper mode is already enabled for one of the timer outputs.

**Reference Manual to  
LL API cross  
reference:**

- CHPxR CARFRQ LL\_HRTIM\_CHP\_SetPrescaler

**LL\_HRTIM\_CHP\_GetPrescaler**

**Function name** `__STATIC_INLINE uint32_t LL_HRTIM_CHP_GetPrescaler(  
HRTIM_TypeDef * HRTIMx, uint32_t Timer)`

**Function description** Get actual chopper stage prescaler value.

**Parameters**

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

**Return values**

- **Prescaler:** This parameter can be one of the following values:
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV16
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV32
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV48
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV64
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV80
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV96
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV112
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV128
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV144
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV160
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV176
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV192
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV208
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV224
  - LL\_HRTIM\_CHP\_PRESCALER\_DIV240

Reference Manual to  
LL API cross  
reference:

- LL\_HRTIM\_CHP\_PRESCALER\_DIV256

- CHPxR CARFRQ LL\_HRTIM\_CHP\_GetPrescaler

### **LL\_HRTIM\_CHP\_SetDutyCycle**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_CHP_SetDutyCycle(<br/>HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t<br/>DutyCycle)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Set the chopper duty cycle.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> <li>• <b>DutyCycle:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_CHP_DUTYCYCLE_0</li> <li>- LL_HRTIM_CHP_DUTYCYCLE_125</li> <li>- LL_HRTIM_CHP_DUTYCYCLE_250</li> <li>- LL_HRTIM_CHP_DUTYCYCLE_375</li> <li>- LL_HRTIM_CHP_DUTYCYCLE_500</li> <li>- LL_HRTIM_CHP_DUTYCYCLE_625</li> <li>- LL_HRTIM_CHP_DUTYCYCLE_750</li> <li>- LL_HRTIM_CHP_DUTYCYCLE_875</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Duty cycle can be adjusted by 1/8 step (from 0/8 up to 7/8)</li> <li>• This function must not be called if the chopper mode is already enabled for one of the timer outputs.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CHPxR CARDTY LL_HRTIM_CHP_SetDutyCycle</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

### **LL\_HRTIM\_CHP\_GetDutyCycle**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_CHP_GetDutyCycle(<br/>HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                        |
| Function description | Get actual chopper duty cycle.                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Return values</b>                              | <ul style="list-style-type: none"> <li>• <b>DutyCycle:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_CHP_DUTYCYCLE_0</li> <li>– LL_HRTIM_CHP_DUTYCYCLE_125</li> <li>– LL_HRTIM_CHP_DUTYCYCLE_250</li> <li>– LL_HRTIM_CHP_DUTYCYCLE_375</li> <li>– LL_HRTIM_CHP_DUTYCYCLE_500</li> <li>– LL_HRTIM_CHP_DUTYCYCLE_625</li> <li>– LL_HRTIM_CHP_DUTYCYCLE_750</li> <li>– LL_HRTIM_CHP_DUTYCYCLE_875</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CHPxR CARDTY LL_HRTIM_CHP_GetDutyCycle</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                 |

### LL\_HRTIM\_CHP\_SetPulseWidth

Function name      **`__STATIC_INLINE void LL_HRTIM_CHP_SetPulseWidth(HRTIM_TypeDef * HRTIMx, uint32_t Timer, uint32_t PulseWidth)`**

Function description      Set the start pulse width.

Parameters     
 

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E
- **PulseWidth:** This parameter can be one of the following values:
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_16
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_32
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_48
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_64
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_80
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_96
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_112
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_128
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_144
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_160
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_176
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_192
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_208
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_224
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_240
  - LL\_HRTIM\_CHP\_PULSEWIDTH\_256

Return values     
 

- **None**

Notes     
 

- This function must not be called if the chopper mode is already enabled for one of the timer outputs.

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CHPxR STRPW LL_HRTIM_CHP_SetPulseWidth</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_CHP_GetPulseWidth(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description                              | Get actual start pulse width.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>PulseWidth:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_HRTIM_CHP_PULSEWIDTH_16</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_32</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_48</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_64</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_80</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_96</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_112</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_128</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_144</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_160</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_176</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_192</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_208</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_224</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_240</li> <li>- LL_HRTIM_CHP_PULSEWIDTH_256</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CHPxR STRPW LL_HRTIM_CHP_GetPulseWidth</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

### LL\_HRTIM\_OUT\_SetOutputSetSrc

|                      |                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_OUT_SetOutputSetSrc(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Output, uint32_t SetSrc)</code>                                                                                                                                                                                                                                                               |
| Function description | Set the timer output set source.                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> </ul> </li> </ul> |

- LL\_HRTIM\_OUTPUT\_TC2
- LL\_HRTIM\_OUTPUT\_TD1
- LL\_HRTIM\_OUTPUT\_TD2
- LL\_HRTIM\_OUTPUT\_TE1
- LL\_HRTIM\_OUTPUT\_TE2
- **SetSrc:** This parameter can be a combination of the following values:
  - LL\_HRTIM\_CROSSBAR\_NONE
  - LL\_HRTIM\_CROSSBAR\_RESYNC
  - LL\_HRTIM\_CROSSBAR\_TIMPER
  - LL\_HRTIM\_CROSSBAR\_TIMCMP1
  - LL\_HRTIM\_CROSSBAR\_TIMCMP2
  - LL\_HRTIM\_CROSSBAR\_TIMCMP3
  - LL\_HRTIM\_CROSSBAR\_TIMCMP4
  - LL\_HRTIM\_CROSSBAR\_MASTERPER
  - LL\_HRTIM\_CROSSBAR\_MASTERCMP1
  - LL\_HRTIM\_CROSSBAR\_MASTERCMP2
  - LL\_HRTIM\_CROSSBAR\_MASTERCMP3
  - LL\_HRTIM\_CROSSBAR\_MASTERCMP4
  - LL\_HRTIM\_CROSSBAR\_TIMEV\_1
  - LL\_HRTIM\_CROSSBAR\_TIMEV\_2
  - LL\_HRTIM\_CROSSBAR\_TIMEV\_3
  - LL\_HRTIM\_CROSSBAR\_TIMEV\_4
  - LL\_HRTIM\_CROSSBAR\_TIMEV\_5
  - LL\_HRTIM\_CROSSBAR\_TIMEV\_6
  - LL\_HRTIM\_CROSSBAR\_TIMEV\_7
  - LL\_HRTIM\_CROSSBAR\_TIMEV\_8
  - LL\_HRTIM\_CROSSBAR\_TIMEV\_9
  - LL\_HRTIM\_CROSSBAR\_EEV\_1
  - LL\_HRTIM\_CROSSBAR\_EEV\_2
  - LL\_HRTIM\_CROSSBAR\_EEV\_3
  - LL\_HRTIM\_CROSSBAR\_EEV\_4
  - LL\_HRTIM\_CROSSBAR\_EEV\_5
  - LL\_HRTIM\_CROSSBAR\_EEV\_6
  - LL\_HRTIM\_CROSSBAR\_EEV\_7
  - LL\_HRTIM\_CROSSBAR\_EEV\_8
  - LL\_HRTIM\_CROSSBAR\_EEV\_9
  - LL\_HRTIM\_CROSSBAR\_EEV\_10
  - LL\_HRTIM\_CROSSBAR\_UPDATE

**Return values**

- **None**
- SETx1R SST LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R RESYNC LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R PER LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R CMP1 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R CMP2 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R CMP3 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R CMP4 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R MSTPER LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R MSTCMP1 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R MSTCMP2 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R MSTCMP3 LL\_HRTIM\_OUT\_SetOutputSetSrc

- SETx1R MSTCMP4 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT1 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT2 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT3 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT4 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT5 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT6 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT7 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT8 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT9 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT1 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT2 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT3 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT4 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT5 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT6 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT7 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT8 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT9 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT10 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R UPDATE LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R SST LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R RESYNC LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R PER LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R CMP1 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R CMP2 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R CMP3 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R CMP4 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R MSTPER LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R MSTCMP1 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R MSTCMP2 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R MSTCMP3 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R MSTCMP4 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT1 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT2 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT3 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT4 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT5 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT6 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT7 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT8 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT9 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT1 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT2 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT3 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT4 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT5 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT6 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT7 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT8 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT9 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT10 LL\_HRTIM\_OUT\_SetOutputSetSrc

- SETx1R UPDATE LL\_HRTIM\_OUT\_SetOutputSetSrc

### **LL\_HRTIM\_OUT\_GetOutputSetSrc**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_HRTIM_OUT_GetOutputSetSrc(<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Output)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description | Get the timer output set source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Return values        | <ul style="list-style-type: none"> <li>• <b>SetSrc:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_CROSSBAR_NONE</li> <li>- LL_HRTIM_CROSSBAR_RESYNC</li> <li>- LL_HRTIM_CROSSBAR_TIMPER</li> <li>- LL_HRTIM_CROSSBAR_TIMCMP1</li> <li>- LL_HRTIM_CROSSBAR_TIMCMP2</li> <li>- LL_HRTIM_CROSSBAR_TIMCMP3</li> <li>- LL_HRTIM_CROSSBAR_TIMCMP4</li> <li>- LL_HRTIM_CROSSBAR_MASTERPER</li> <li>- LL_HRTIM_CROSSBAR_MASTERCMP1</li> <li>- LL_HRTIM_CROSSBAR_MASTERCMP2</li> <li>- LL_HRTIM_CROSSBAR_MASTERCMP3</li> <li>- LL_HRTIM_CROSSBAR_MASTERCMP4</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_1</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_2</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_3</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_4</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_5</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_6</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_7</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_8</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_9</li> <li>- LL_HRTIM_CROSSBAR_EEV_1</li> <li>- LL_HRTIM_CROSSBAR_EEV_2</li> <li>- LL_HRTIM_CROSSBAR_EEV_3</li> <li>- LL_HRTIM_CROSSBAR_EEV_4</li> <li>- LL_HRTIM_CROSSBAR_EEV_5</li> <li>- LL_HRTIM_CROSSBAR_EEV_6</li> <li>- LL_HRTIM_CROSSBAR_EEV_7</li> <li>- LL_HRTIM_CROSSBAR_EEV_8</li> <li>- LL_HRTIM_CROSSBAR_EEV_9</li> </ul> </li> </ul> |

Reference Manual to  
LL API cross  
reference:

- LL\_HRTIM\_CROSSBAR\_EEV\_10
- LL\_HRTIM\_CROSSBAR\_UPDATE
- SETx1R SST LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R RESYNC LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R PER LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R CMP1 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R CMP2 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R CMP3 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R CMP4 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R MSTPER LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R MSTCMP1 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R MSTCMP2 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R MSTCMP3 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R MSTCMP4 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT1 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT2 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT3 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT4 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT5 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT6 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT7 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT8 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT9 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R EXEVNT1 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R EXEVNT2 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R EXEVNT3 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R EXEVNT4 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R EXEVNT5 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R EXEVNT6 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R EXEVNT7 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R EXEVNT8 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R EXEVNT9 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R EXEVNT10 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R UPDATE LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R SST LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R RESYNC LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R PER LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R CMP1 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R CMP2 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R CMP3 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R CMP4 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R MSTPER LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R MSTCMP1 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R MSTCMP2 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R MSTCMP3 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R MSTCMP4 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT1 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT2 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT3 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT4 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT5 LL\_HRTIM\_OUT\_GetOutputSetSrc
- SETx1R TIMEVNT6 LL\_HRTIM\_OUT\_GetOutputSetSrc

- SETx1R TIMEVNT7 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT8 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R TIMEVNT9 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT1 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT2 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT3 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT4 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT5 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT6 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT7 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT8 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT9 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R EXEVNT10 LL\_HRTIM\_OUT\_SetOutputSetSrc
- SETx1R UPDATE LL\_HRTIM\_OUT\_SetOutputSetSrc

### **LL\_HRTIM\_OUT\_SetOutputResetSrc**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>STATIC_INLINE void LL_HRTIM_OUT_SetOutputResetSrc(<br/>HRTIM_TypeDef * HRTIMx, uint32_t Output, uint32_t<br/>ResetSrc)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Set the timer output reset source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> </li> <li>• <b>ResetSrc:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_CROSSBAR_NONE</li> <li>- LL_HRTIM_CROSSBAR_RESYNC</li> <li>- LL_HRTIM_CROSSBAR_TIMPER</li> <li>- LL_HRTIM_CROSSBAR_TIMCMP1</li> <li>- LL_HRTIM_CROSSBAR_TIMCMP2</li> <li>- LL_HRTIM_CROSSBAR_TIMCMP3</li> <li>- LL_HRTIM_CROSSBAR_TIMCMP4</li> <li>- LL_HRTIM_CROSSBAR_MASTERPER</li> <li>- LL_HRTIM_CROSSBAR_MASTERCMP1</li> <li>- LL_HRTIM_CROSSBAR_MASTERCMP2</li> <li>- LL_HRTIM_CROSSBAR_MASTERCMP3</li> <li>- LL_HRTIM_CROSSBAR_MASTERCMP4</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_1</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_2</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_3</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_4</li> <li>- LL_HRTIM_CROSSBAR_TIMEV_5</li> </ul> </li> </ul> |

- LL\_HRTIM\_CROSSBAR\_TIMEV\_6
- LL\_HRTIM\_CROSSBAR\_TIMEV\_7
- LL\_HRTIM\_CROSSBAR\_TIMEV\_8
- LL\_HRTIM\_CROSSBAR\_TIMEV\_9
- LL\_HRTIM\_CROSSBAR\_EEV\_1
- LL\_HRTIM\_CROSSBAR\_EEV\_2
- LL\_HRTIM\_CROSSBAR\_EEV\_3
- LL\_HRTIM\_CROSSBAR\_EEV\_4
- LL\_HRTIM\_CROSSBAR\_EEV\_5
- LL\_HRTIM\_CROSSBAR\_EEV\_6
- LL\_HRTIM\_CROSSBAR\_EEV\_7
- LL\_HRTIM\_CROSSBAR\_EEV\_8
- LL\_HRTIM\_CROSSBAR\_EEV\_9
- LL\_HRTIM\_CROSSBAR\_EEV\_10
- LL\_HRTIM\_CROSSBAR\_UPDATE

## Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- RSTx1R RST LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R RESYNC LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R PER LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R CMP1 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R CMP2 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R CMP3 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R CMP4 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R MSTPER LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R MSTCMP1 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R MSTCMP2 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R MSTCMP3 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R MSTCMP4 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT1 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT2 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT3 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT4 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT5 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT6 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT7 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT8 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT9 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT1 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT2 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT3 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT4 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT5 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT6 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT7 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT8 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT9 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT10 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R UPDATE LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R RST LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R RESYNC LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R PER LL\_HRTIM\_OUT\_SetOutputResetSrc

- RSTx1R CMP1 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R CMP2 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R CMP3 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R CMP4 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R MSTPER LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R MSTCMP1 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R MSTCMP2 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R MSTCMP3 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R MSTCMP4 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT1 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT2 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT3 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT4 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT5 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT6 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT7 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT8 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R TIMEVNT9 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT1 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT2 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT3 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT4 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT5 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT6 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT7 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT8 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT9 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R EXEVNT10 LL\_HRTIM\_OUT\_SetOutputResetSrc
- RSTx1R UPDATE LL\_HRTIM\_OUT\_SetOutputResetSrc

### **LL\_HRTIM\_OUT\_GetOutputResetSrc**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_OUT_GetOutputResetSrc (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Output)</code>                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description | Get the timer output set source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>ResetSrc:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_CROSSBAR_NONE</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                  |

- LL\_HRTIM\_CROSSBAR\_RESETSRC
- LL\_HRTIM\_CROSSBAR\_TIMPER
- LL\_HRTIM\_CROSSBAR\_TIMCMP1
- LL\_HRTIM\_CROSSBAR\_TIMCMP2
- LL\_HRTIM\_CROSSBAR\_TIMCMP3
- LL\_HRTIM\_CROSSBAR\_TIMCMP4
- LL\_HRTIM\_CROSSBAR\_MASTERPER
- LL\_HRTIM\_CROSSBAR\_MASTERCMP1
- LL\_HRTIM\_CROSSBAR\_MASTERCMP2
- LL\_HRTIM\_CROSSBAR\_MASTERCMP3
- LL\_HRTIM\_CROSSBAR\_MASTERCMP4
- LL\_HRTIM\_CROSSBAR\_TIMEV\_1
- LL\_HRTIM\_CROSSBAR\_TIMEV\_2
- LL\_HRTIM\_CROSSBAR\_TIMEV\_3
- LL\_HRTIM\_CROSSBAR\_TIMEV\_4
- LL\_HRTIM\_CROSSBAR\_TIMEV\_5
- LL\_HRTIM\_CROSSBAR\_TIMEV\_6
- LL\_HRTIM\_CROSSBAR\_TIMEV\_7
- LL\_HRTIM\_CROSSBAR\_TIMEV\_8
- LL\_HRTIM\_CROSSBAR\_TIMEV\_9
- LL\_HRTIM\_CROSSBAR\_EEV\_1
- LL\_HRTIM\_CROSSBAR\_EEV\_2
- LL\_HRTIM\_CROSSBAR\_EEV\_3
- LL\_HRTIM\_CROSSBAR\_EEV\_4
- LL\_HRTIM\_CROSSBAR\_EEV\_5
- LL\_HRTIM\_CROSSBAR\_EEV\_6
- LL\_HRTIM\_CROSSBAR\_EEV\_7
- LL\_HRTIM\_CROSSBAR\_EEV\_8
- LL\_HRTIM\_CROSSBAR\_EEV\_9
- LL\_HRTIM\_CROSSBAR\_EEV\_10
- LL\_HRTIM\_CROSSBAR\_UPDATE

Reference Manual to  
LL API cross  
reference:

- RSTx1R RST LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R RESYNC LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R PER LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R CMP1 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R CMP2 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R CMP3 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R CMP4 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R MSTPER LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R MSTCMP1 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R MSTCMP2 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R MSTCMP3 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R MSTCMP4 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT1 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT2 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT3 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT4 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT5 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT6 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT7 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT8 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT9 LL\_HRTIM\_OUT\_GetOutputResetSrc

- RSTx1R EXEVNT1 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT2 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT3 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT4 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT5 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT6 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT7 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT8 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT9 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT10 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R UPDATE LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R RST LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R RESYNC LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R PER LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R CMP1 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R CMP2 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R CMP3 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R CMP4 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R MSTPER LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R MSTCMP1 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R MSTCMP2 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R MSTCMP3 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R MSTCMP4 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT1 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT2 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT3 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT4 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT5 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT6 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT7 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT8 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R TIMEVNT9 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT1 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT2 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT3 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT4 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT5 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT6 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT7 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT8 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT9 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R EXEVNT10 LL\_HRTIM\_OUT\_GetOutputResetSrc
- RSTx1R UPDATE LL\_HRTIM\_OUT\_GetOutputResetSrc

### LL\_HRTIM\_OUT\_Config

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_OUT_Config<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Output, uint32_t Configuration)</code>                                                    |
| Function description | Configure a timer output.                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values:</li> </ul> |

- LL\_HRTIM\_OUTPUT\_TA1
- LL\_HRTIM\_OUTPUT\_TA2
- LL\_HRTIM\_OUTPUT\_TB1
- LL\_HRTIM\_OUTPUT\_TB2
- LL\_HRTIM\_OUTPUT\_TC1
- LL\_HRTIM\_OUTPUT\_TC2
- LL\_HRTIM\_OUTPUT\_TD1
- LL\_HRTIM\_OUTPUT\_TD2
- LL\_HRTIM\_OUTPUT\_TE1
- LL\_HRTIM\_OUTPUT\_TE2
- **Configuration:** This parameter must be a combination of all the following values:
  - LL\_HRTIM\_OUT\_POSITIVE\_POLARITY or LL\_HRTIM\_OUT\_NEGATIVE\_POLARITY
  - LL\_HRTIM\_OUT\_NO\_IDLE or LL\_HRTIM\_OUT\_IDLE\_WHEN\_BURST
  - LL\_HRTIM\_OUT\_IDLELEVEL\_INACTIVE or LL\_HRTIM\_OUT\_IDLELEVEL\_ACTIVE
  - LL\_HRTIM\_OUT\_FAULTSTATE\_NO\_ACTION or LL\_HRTIM\_OUT\_FAULTSTATE\_ACTIVE or LL\_HRTIM\_OUT\_FAULTSTATE\_INACTIVE or LL\_HRTIM\_OUT\_FAULTSTATE\_HIGHZ
  - LL\_HRTIM\_OUT\_CHOPPERMODE\_DISABLED or LL\_HRTIM\_OUT\_CHOPPERMODE\_ENABLED
  - LL\_HRTIM\_OUT\_BM\_ENTRYMODE\_REGULAR or LL\_HRTIM\_OUT\_BM\_ENTRYMODE\_DELAYED

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- OUTxR POL1 LL\_HRTIM\_OUT\_Config
- OUTxR IDLEM1 LL\_HRTIM\_OUT\_Config
- OUTxR IDLES1 LL\_HRTIM\_OUT\_Config
- OUTxR FAULT1 LL\_HRTIM\_OUT\_Config
- OUTxR CHP1 LL\_HRTIM\_OUT\_Config
- OUTxR DIDL1 LL\_HRTIM\_OUT\_Config
- OUTxR POL2 LL\_HRTIM\_OUT\_Config
- OUTxR IDLEM2 LL\_HRTIM\_OUT\_Config
- OUTxR IDLES2 LL\_HRTIM\_OUT\_Config
- OUTxR FAULT2 LL\_HRTIM\_OUT\_Config
- OUTxR CHP2 LL\_HRTIM\_OUT\_Config
- OUTxR DIDL2 LL\_HRTIM\_OUT\_Config

### **LL\_HRTIM\_OUT\_SetPolarity**

Function name

**\_STATIC\_INLINE void LL\_HRTIM\_OUT\_SetPolarity(  
(HRTIM\_TypeDef \* HRTIMx, uint32\_t Output, uint32\_t Polarity)**

Function description

Set the polarity of a timer output.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Output:** This parameter can be one of the following values:
  - LL\_HRTIM\_OUTPUT\_TA1
  - LL\_HRTIM\_OUTPUT\_TA2
  - LL\_HRTIM\_OUTPUT\_TB1
  - LL\_HRTIM\_OUTPUT\_TB2

- LL\_HRTIM\_OUTPUT\_TC1
- LL\_HRTIM\_OUTPUT\_TC2
- LL\_HRTIM\_OUTPUT\_TD1
- LL\_HRTIM\_OUTPUT\_TD2
- LL\_HRTIM\_OUTPUT\_TE1
- LL\_HRTIM\_OUTPUT\_TE2
- **Polarity:** This parameter can be one of the following values:
  - LL\_HRTIM\_OUT\_POSITIVE\_POLARITY
  - LL\_HRTIM\_OUT\_NEGATIVE\_POLARITY

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- OUTxR POL1 LL\_HRTIM\_OUT\_SetPolarity
- OUTxR POL2 LL\_HRTIM\_OUT\_SetPolarity

**LL\_HRTIM\_OUT\_GetPolarity**

Function name

```
_STATIC_INLINE uint32_t LL_HRTIM_OUT_GetPolarity
(HRTIM_TypeDef * HRTIMx, uint32_t Output)
```

Function description

Get actual polarity of the timer output.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Output:** This parameter can be one of the following values:
  - LL\_HRTIM\_OUTPUT\_TA1
  - LL\_HRTIM\_OUTPUT\_TA2
  - LL\_HRTIM\_OUTPUT\_TB1
  - LL\_HRTIM\_OUTPUT\_TB2
  - LL\_HRTIM\_OUTPUT\_TC1
  - LL\_HRTIM\_OUTPUT\_TC2
  - LL\_HRTIM\_OUTPUT\_TD1
  - LL\_HRTIM\_OUTPUT\_TD2
  - LL\_HRTIM\_OUTPUT\_TE1
  - LL\_HRTIM\_OUTPUT\_TE2

Return values

- **Polarity:** This parameter can be one of the following values:
  - LL\_HRTIM\_OUT\_POSITIVE\_POLARITY
  - LL\_HRTIM\_OUT\_NEGATIVE\_POLARITY

Reference Manual to  
LL API cross  
reference:

- OUTxR POL1 LL\_HRTIM\_OUT\_SetPolarity
- OUTxR POL2 LL\_HRTIM\_OUT\_SetPolarity

**LL\_HRTIM\_OUT\_SetIdleMode**

Function name

```
_STATIC_INLINE void LL_HRTIM_OUT_SetIdleMode
(HRTIM_TypeDef * HRTIMx, uint32_t Output, uint32_t
IdleMode)
```

Function description

Set the output IDLE mode.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Output:** This parameter can be one of the following values:
  - LL\_HRTIM\_OUTPUT\_TA1
  - LL\_HRTIM\_OUTPUT\_TA2
  - LL\_HRTIM\_OUTPUT\_TB1

|                                                   |                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> |
|                                                   | <ul style="list-style-type: none"> <li>• <b>IdleMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUT_NO_IDLE</li> <li>- LL_HRTIM_OUT_IDLE_WHEN_BURST</li> </ul> </li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the burst mode is active</li> </ul>                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR IDLEM1 LL_HRTIM_OUT_SetIdleMode</li> <li>• OUTxR IDLEM2 LL_HRTIM_OUT_SetIdleMode</li> </ul>                                                                                                                        |

### LL\_HRTIM\_OUT\_GetIdleMode

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_OUT_GetIdleMode(<br/>  HRTIM_TypeDef * HRTIMx, uint32_t Output)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Get actual output IDLE mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>IdleMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUT_NO_IDLE</li> <li>- LL_HRTIM_OUT_IDLE_WHEN_BURST</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR IDLEM1 LL_HRTIM_OUT_SetIdleMode</li> <li>• OUTxR IDLEM2 LL_HRTIM_OUT_SetIdleMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                               |

### LL\_HRTIM\_OUT\_SetIdleLevel

|                      |                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_OUT_SetIdleLevel(<br/>  HRTIM_TypeDef * HRTIMx, uint32_t Output, uint32_t<br/>  IdleLevel)</code> |
| Function description | Set the output IDLE level.                                                                                                            |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_OUTPUT_TA1</li> <li>– LL_HRTIM_OUTPUT_TA2</li> <li>– LL_HRTIM_OUTPUT_TB1</li> <li>– LL_HRTIM_OUTPUT_TB2</li> <li>– LL_HRTIM_OUTPUT_TC1</li> <li>– LL_HRTIM_OUTPUT_TC2</li> <li>– LL_HRTIM_OUTPUT_TD1</li> <li>– LL_HRTIM_OUTPUT_TD2</li> <li>– LL_HRTIM_OUTPUT_TE1</li> <li>– LL_HRTIM_OUTPUT_TE2</li> </ul> </li> <li>• <b>IdleLevel:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_OUT_IDLELEVEL_INACTIVE</li> <li>– LL_HRTIM_OUT_IDLELEVEL_ACTIVE</li> </ul> </li> </ul> |
| Return values                                     | • <b>None</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must be called prior enabling the timer.</li> <li>• Idle level isn't relevant when the output idle mode is set to LL_HRTIM_OUT_NO_IDLE.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR IDLES1 LL_HRTIM_OUT_SetIdleLevel</li> <li>• OUTxR IDLES2 LL_HRTIM_OUT_SetIdleLevel</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

## LL\_HRTIM\_OUT\_GetIdleLevel

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_OUT_GetIdleLevel(<br/>  HRTIM_TypeDef * HRTIMx, uint32_t Output)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Get actual output IDLE level.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_OUTPUT_TA1</li> <li>– LL_HRTIM_OUTPUT_TA2</li> <li>– LL_HRTIM_OUTPUT_TB1</li> <li>– LL_HRTIM_OUTPUT_TB2</li> <li>– LL_HRTIM_OUTPUT_TC1</li> <li>– LL_HRTIM_OUTPUT_TC2</li> <li>– LL_HRTIM_OUTPUT_TD1</li> <li>– LL_HRTIM_OUTPUT_TD2</li> <li>– LL_HRTIM_OUTPUT_TE1</li> <li>– LL_HRTIM_OUTPUT_TE2</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>IdleLevel:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_OUT_IDLELEVEL_INACTIVE</li> <li>– LL_HRTIM_OUT_IDLELEVEL_ACTIVE</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR IDLES1 LL_HRTIM_OUT_SetIdleLevel</li> <li>• OUTxR IDLES2 LL_HRTIM_OUT_SetIdleLevel</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                             |

**LL\_HRTIM\_OUT\_SetFaultState**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_OUT_SetFaultState<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Output, uint32_t<br/>FaultState)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description                              | Set the output FAULT state.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_OUTPUT_TA1</li> <li>– LL_HRTIM_OUTPUT_TA2</li> <li>– LL_HRTIM_OUTPUT_TB1</li> <li>– LL_HRTIM_OUTPUT_TB2</li> <li>– LL_HRTIM_OUTPUT_TC1</li> <li>– LL_HRTIM_OUTPUT_TC2</li> <li>– LL_HRTIM_OUTPUT_TD1</li> <li>– LL_HRTIM_OUTPUT_TD2</li> <li>– LL_HRTIM_OUTPUT_TE1</li> <li>– LL_HRTIM_OUTPUT_TE2</li> </ul> </li> <li>• <b>FaultState:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_OUT_FAULTSTATE_NO_ACTION</li> <li>– LL_HRTIM_OUT_FAULTSTATE_ACTIVE</li> <li>– LL_HRTIM_OUT_FAULTSTATE_INACTIVE</li> <li>– LL_HRTIM_OUT_FAULTSTATE_HIGHZ</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the timer is enabled and a fault channel is enabled at timer level.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR FAULT1 LL_HRTIM_OUT_SetFaultState</li> <li>• OUTxR FAULT2 LL_HRTIM_OUT_SetFaultState</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

**LL\_HRTIM\_OUT\_GetFaultState**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_OUT_GetFaultState<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Output)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Get actual FAULT state.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_OUTPUT_TA1</li> <li>– LL_HRTIM_OUTPUT_TA2</li> <li>– LL_HRTIM_OUTPUT_TB1</li> <li>– LL_HRTIM_OUTPUT_TB2</li> <li>– LL_HRTIM_OUTPUT_TC1</li> <li>– LL_HRTIM_OUTPUT_TC2</li> <li>– LL_HRTIM_OUTPUT_TD1</li> <li>– LL_HRTIM_OUTPUT_TD2</li> <li>– LL_HRTIM_OUTPUT_TE1</li> <li>– LL_HRTIM_OUTPUT_TE2</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>FaultState:</b> This parameter can be one of the following values:</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                      |

- LL\_HRTIM\_OUT\_FAULTSTATE\_NO\_ACTION
- LL\_HRTIM\_OUT\_FAULTSTATE\_ACTIVE
- LL\_HRTIM\_OUT\_FAULTSTATE\_INACTIVE
- LL\_HRTIM\_OUT\_FAULTSTATE\_HIGHZ

Reference Manual to  
LL API cross  
reference:

- OUTxR FAULT1 LL\_HRTIM\_OUT\_GetFaultState
- OUTxR FAULT2 LL\_HRTIM\_OUT\_GetFaultState

### **LL\_HRTIM\_OUT\_SetChopperMode**

Function name      **`__STATIC_INLINE void LL_HRTIM_OUT_SetChopperMode(HRTIM_TypeDef * HRTIMx, uint32_t Output, uint32_t ChopperMode)`**

Function description      Set the output chopper mode.

- Parameters
- **HRTIMx:** High Resolution Timer instance
  - **Output:** This parameter can be one of the following values:
    - LL\_HRTIM\_OUTPUT\_TA1
    - LL\_HRTIM\_OUTPUT\_TA2
    - LL\_HRTIM\_OUTPUT\_TB1
    - LL\_HRTIM\_OUTPUT\_TB2
    - LL\_HRTIM\_OUTPUT\_TC1
    - LL\_HRTIM\_OUTPUT\_TC2
    - LL\_HRTIM\_OUTPUT\_TD1
    - LL\_HRTIM\_OUTPUT\_TD2
    - LL\_HRTIM\_OUTPUT\_TE1
    - LL\_HRTIM\_OUTPUT\_TE2
  - **ChopperMode:** This parameter can be one of the following values:
    - LL\_HRTIM\_OUT\_CHOPPERMODE\_DISABLED
    - LL\_HRTIM\_OUT\_CHOPPERMODE\_ENABLED

Return values      • **None**

Notes      • This function must not be called when the timer is enabled.

Reference Manual to  
LL API cross  
reference:

- OUTxR CHP1 LL\_HRTIM\_OUT\_SetChopperMode
- OUTxR CHP2 LL\_HRTIM\_OUT\_SetChopperMode

### **LL\_HRTIM\_OUT\_GetChopperMode**

Function name      **`__STATIC_INLINE uint32_t LL_HRTIM_OUT_GetChopperMode(HRTIM_TypeDef * HRTIMx, uint32_t Output)`**

Function description      Get actual output chopper mode.

- Parameters
- **HRTIMx:** High Resolution Timer instance
  - **Output:** This parameter can be one of the following values:
    - LL\_HRTIM\_OUTPUT\_TA1
    - LL\_HRTIM\_OUTPUT\_TA2
    - LL\_HRTIM\_OUTPUT\_TB1
    - LL\_HRTIM\_OUTPUT\_TB2
    - LL\_HRTIM\_OUTPUT\_TC1
    - LL\_HRTIM\_OUTPUT\_TC2

|                                                   |                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul>                                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>ChopperMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUT_CHOPPERMODE_DISABLED</li> <li>- LL_HRTIM_OUT_CHOPPERMODE_ENABLED</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR CHP1 LL_HRTIM_OUT_GetChopperMode</li> <li>• OUTxR CHP2 LL_HRTIM_OUT_GetChopperMode</li> </ul>                                                                                                                            |

### LL\_HRTIM\_OUT\_SetBMEEntryMode

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_OUT_SetBMEEntryMode(HRTIM_TypeDef * HRTIMx, uint32_t Output, uint32_t BMEEntryMode)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Set the output burst mode entry mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> </li> <li>• <b>BMEEntryMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUT_BM_ENTRYMODE_REGULAR</li> <li>- LL_HRTIM_OUT_BM_ENTRYMODE_DELAYED</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the timer is enabled.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR DIDL1 LL_HRTIM_OUT_SetBMEEntryMode</li> <li>• OUTxR DIDL2 LL_HRTIM_OUT_SetBMEEntryMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

### LL\_HRTIM\_OUT\_GetBMEEntryMode

|                      |                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_OUT_GetBMEEntryMode(HRTIM_TypeDef * HRTIMx, uint32_t Output)</code>                                                                                                                                                                                      |
| Function description | Get actual output burst mode entry mode.                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>BMEEntryMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUT_BM_ENTRYMODE_REGULAR</li> <li>- LL_HRTIM_OUT_BM_ENTRYMODE_DELAYED</li> </ul> </li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OUTxR DIDL1 LL_HRTIM_OUT_GetBMEEntryMode</li> <li>• OUTxR DIDL2 LL_HRTIM_OUT_GetBMEEntryMode</li> </ul>                                                                                                                                                 |

### LL\_HRTIM\_OUT\_GetDLYPRTOutStatus

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_OUT_GetDLYPRTOutStatus (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Output)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description                              | Get the level (active or inactive) of the designated output when the delayed protection was triggered.                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Output:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUTPUT_TA1</li> <li>- LL_HRTIM_OUTPUT_TA2</li> <li>- LL_HRTIM_OUTPUT_TB1</li> <li>- LL_HRTIM_OUTPUT_TB2</li> <li>- LL_HRTIM_OUTPUT_TC1</li> <li>- LL_HRTIM_OUTPUT_TC2</li> <li>- LL_HRTIM_OUTPUT_TD1</li> <li>- LL_HRTIM_OUTPUT_TD2</li> <li>- LL_HRTIM_OUTPUT_TE1</li> <li>- LL_HRTIM_OUTPUT_TE2</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>OutputLevel:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_OUT_LEVEL_INACTIVE</li> <li>- LL_HRTIM_OUT_LEVEL_ACTIVE</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxISR O1SRSR LL_HRTIM_OUT_GetDLYPRTOutStatus</li> <li>• TIMxISR O2SRSR LL_HRTIM_OUT_GetDLYPRTOutStatus</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                             |

### LL\_HRTIM\_OUT\_ForceLevel

|                      |                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_OUT_ForceLevel<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Output, uint32_t<br/>OutputLevel)</code> |
| Function description | Force the timer output to its active or inactive level.                                                                           |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Output:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_OUTPUT_TA1</li> <li>– LL_HRTIM_OUTPUT_TA2</li> <li>– LL_HRTIM_OUTPUT_TB1</li> <li>– LL_HRTIM_OUTPUT_TB2</li> <li>– LL_HRTIM_OUTPUT_TC1</li> <li>– LL_HRTIM_OUTPUT_TC2</li> <li>– LL_HRTIM_OUTPUT_TD1</li> <li>– LL_HRTIM_OUTPUT_TD2</li> <li>– LL_HRTIM_OUTPUT_TE1</li> <li>– LL_HRTIM_OUTPUT_TE2</li> </ul> </li> <li><b>OutputLevel:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_OUT_LEVEL_INACTIVE</li> <li>– LL_HRTIM_OUT_LEVEL_ACTIVE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SETx1R SST LL_HRTIM_OUT_ForceLevel</li> <li>RSTx1R SRT LL_HRTIM_OUT_ForceLevel</li> <li>SETx2R SST LL_HRTIM_OUT_ForceLevel</li> <li>RSTx2R SRT LL_HRTIM_OUT_ForceLevel</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

### LL\_HRTIM\_OUT\_GetLevel

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_OUT_GetLevel(<br/>  HRTIM_TypeDef * HRTIMx, uint32_t Output)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Get actual output level, before the output stage (chopper, polarity).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Output:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_OUTPUT_TA1</li> <li>– LL_HRTIM_OUTPUT_TA2</li> <li>– LL_HRTIM_OUTPUT_TB1</li> <li>– LL_HRTIM_OUTPUT_TB2</li> <li>– LL_HRTIM_OUTPUT_TC1</li> <li>– LL_HRTIM_OUTPUT_TC2</li> <li>– LL_HRTIM_OUTPUT_TD1</li> <li>– LL_HRTIM_OUTPUT_TD2</li> <li>– LL_HRTIM_OUTPUT_TE1</li> <li>– LL_HRTIM_OUTPUT_TE2</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>OutputLevel:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_OUT_LEVEL_INACTIVE</li> <li>– LL_HRTIM_OUT_LEVEL_ACTIVE</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TIMxISR O1CPY LL_HRTIM_OUT_GetLevel</li> <li>TIMxISR O2CPY LL_HRTIM_OUT_GetLevel</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                   |

**LL\_HRTIM\_EE\_Config**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_HRTIM_EE_Config<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Event, uint32_t<br/>Configuration)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description                              | Configure external event conditioning.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EVENT_1</li> <li>- LL_HRTIM_EVENT_2</li> <li>- LL_HRTIM_EVENT_3</li> <li>- LL_HRTIM_EVENT_4</li> <li>- LL_HRTIM_EVENT_5</li> <li>- LL_HRTIM_EVENT_6</li> <li>- LL_HRTIM_EVENT_7</li> <li>- LL_HRTIM_EVENT_8</li> <li>- LL_HRTIM_EVENT_9</li> <li>- LL_HRTIM_EVENT_10</li> </ul> </li> <li>• <b>Configuration:</b> This parameter must be a combination of all the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EE_SRC_1 or LL_HRTIM_EE_SRC_2 or LL_HRTIM_EE_SRC_3 or LL_HRTIM_EE_SRC_4</li> <li>- LL_HRTIM_EE_POLARITY_HIGH or LL_HRTIM_EE_POLARITY_LOW</li> <li>- LL_HRTIM_EE_SENSITIVITY_LEVEL or LL_HRTIM_EE_SENSITIVITY_RISINGEDGE or LL_HRTIM_EE_SENSITIVITY_FALLINGEDGE or LL_HRTIM_EE_SENSITIVITY_BOTHEDGES</li> <li>- LL_HRTIM_EE_FASTMODE_DISABLE or LL_HRTIM_EE_FASTMODE_ENABLE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the timer counter is enabled.</li> <li>• Event source (EExSrc1..EExSRC4) mapping depends on configured event channel.</li> <li>• Fast mode is available only for LL_HRTIM_EVENT_1..5.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EECR1 EE1SRC LL_HRTIM_EE_Config</li> <li>• EECR1 EE1POL LL_HRTIM_EE_Config</li> <li>• EECR1 EE1SNS LL_HRTIM_EE_Config</li> <li>• EECR1 EE1FAST LL_HRTIM_EE_Config</li> <li>• EECR1 EE2SRC LL_HRTIM_EE_Config</li> <li>• EECR1 EE2POL LL_HRTIM_EE_Config</li> <li>• EECR1 EE2SNS LL_HRTIM_EE_Config</li> <li>• EECR1 EE2FAST LL_HRTIM_EE_Config</li> <li>• EECR1 EE3SRC LL_HRTIM_EE_Config</li> <li>• EECR1 EE3POL LL_HRTIM_EE_Config</li> <li>• EECR1 EE3SNS LL_HRTIM_EE_Config</li> <li>• EECR1 EE3FAST LL_HRTIM_EE_Config</li> <li>• EECR1 EE4SRC LL_HRTIM_EE_Config</li> <li>• EECR1 EE4POL LL_HRTIM_EE_Config</li> <li>• EECR1 EE4SNS LL_HRTIM_EE_Config</li> </ul>                                                                                                                                                                                                                                                                                                                                             |

- EECR1 EE4FAST LL\_HRTIM\_EE\_Config
- EECR1 EE5SRC LL\_HRTIM\_EE\_Config
- EECR1 EE5POL LL\_HRTIM\_EE\_Config
- EECR1 EE5SNS LL\_HRTIM\_EE\_Config
- EECR1 EE5FAST LL\_HRTIM\_EE\_Config
- EECR2 EE6SRC LL\_HRTIM\_EE\_Config
- EECR2 EE6POL LL\_HRTIM\_EE\_Config
- EECR2 EE6SNS LL\_HRTIM\_EE\_Config
- EECR2 EE6FAST LL\_HRTIM\_EE\_Config
- EECR2 EE7SRC LL\_HRTIM\_EE\_Config
- EECR2 EE7POL LL\_HRTIM\_EE\_Config
- EECR2 EE7SNS LL\_HRTIM\_EE\_Config
- EECR2 EE7FAST LL\_HRTIM\_EE\_Config
- EECR2 EE8SRC LL\_HRTIM\_EE\_Config
- EECR2 EE8POL LL\_HRTIM\_EE\_Config
- EECR2 EE8SNS LL\_HRTIM\_EE\_Config
- EECR2 EE8FAST LL\_HRTIM\_EE\_Config
- EECR2 EE9SRC LL\_HRTIM\_EE\_Config
- EECR2 EE9POL LL\_HRTIM\_EE\_Config
- EECR2 EE9SNS LL\_HRTIM\_EE\_Config
- EECR2 EE9FAST LL\_HRTIM\_EE\_Config
- EECR2 EE10SRC LL\_HRTIM\_EE\_Config
- EECR2 EE10POL LL\_HRTIM\_EE\_Config
- EECR2 EE10SNS LL\_HRTIM\_EE\_Config
- EECR2 EE10FAST LL\_HRTIM\_EE\_Config

### **LL\_HRTIM\_EE\_SetSrc**

|                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                       | <b><code>_STATIC_INLINE void LL_HRTIM_EE_SetSrc<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Event, uint32_t Src)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description                | Set the external event source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                          | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EVENT_1</li> <li>- LL_HRTIM_EVENT_2</li> <li>- LL_HRTIM_EVENT_3</li> <li>- LL_HRTIM_EVENT_4</li> <li>- LL_HRTIM_EVENT_5</li> <li>- LL_HRTIM_EVENT_6</li> <li>- LL_HRTIM_EVENT_7</li> <li>- LL_HRTIM_EVENT_8</li> <li>- LL_HRTIM_EVENT_9</li> <li>- LL_HRTIM_EVENT_10</li> </ul> </li> <li>• <b>Src:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EE_SRC_1</li> <li>- LL_HRTIM_EE_SRC_2</li> <li>- LL_HRTIM_EE_SRC_3</li> <li>- LL_HRTIM_EE_SRC_4</li> </ul> </li> </ul> |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• EECR1 EE1SRC LL_HRTIM_EE_SetSrc</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

- reference:
- EECR1 EE2SRC LL\_HRTIM\_EE\_SetSrc
  - EECR1 EE3SRC LL\_HRTIM\_EE\_SetSrc
  - EECR1 EE4SRC LL\_HRTIM\_EE\_SetSrc
  - EECR1 EE5SRC LL\_HRTIM\_EE\_SetSrc
  - EECR2 EE6SRC LL\_HRTIM\_EE\_SetSrc
  - EECR2 EE7SRC LL\_HRTIM\_EE\_SetSrc
  - EECR2 EE8SRC LL\_HRTIM\_EE\_SetSrc
  - EECR2 EE9SRC LL\_HRTIM\_EE\_SetSrc
  - EECR2 EE10SRC LL\_HRTIM\_EE\_SetSrc

### **LL\_HRTIM\_EE\_GetSrc**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_EE_GetSrc<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Event)</code>                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Get actual external event source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EVENT_1</li> <li>- LL_HRTIM_EVENT_2</li> <li>- LL_HRTIM_EVENT_3</li> <li>- LL_HRTIM_EVENT_4</li> <li>- LL_HRTIM_EVENT_5</li> <li>- LL_HRTIM_EVENT_6</li> <li>- LL_HRTIM_EVENT_7</li> <li>- LL_HRTIM_EVENT_8</li> <li>- LL_HRTIM_EVENT_9</li> <li>- LL_HRTIM_EVENT_10</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>EventSrc:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EE_SRC_1</li> <li>- LL_HRTIM_EE_SRC_2</li> <li>- LL_HRTIM_EE_SRC_3</li> <li>- LL_HRTIM_EE_SRC_4</li> </ul> </li> </ul>                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EECR1 EE1SRC LL_HRTIM_EE_SetSrc</li> <li>• EECR1 EE2SRC LL_HRTIM_EE_SetSrc</li> <li>• EECR1 EE3SRC LL_HRTIM_EE_SetSrc</li> <li>• EECR1 EE4SRC LL_HRTIM_EE_SetSrc</li> <li>• EECR1 EE5SRC LL_HRTIM_EE_SetSrc</li> <li>• EECR2 EE6SRC LL_HRTIM_EE_SetSrc</li> <li>• EECR2 EE7SRC LL_HRTIM_EE_SetSrc</li> <li>• EECR2 EE8SRC LL_HRTIM_EE_SetSrc</li> <li>• EECR2 EE9SRC LL_HRTIM_EE_SetSrc</li> <li>• EECR2 EE10SRC LL_HRTIM_EE_SetSrc</li> </ul>                          |

### **LL\_HRTIM\_EE\_SetPolarity**

|                      |                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_EE_SetPolarity<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Event, uint32_t Polarity)</code> |
| Function description | Set the polarity of an external event.                                                                                    |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_EVENT_1</li> <li>– LL_HRTIM_EVENT_2</li> <li>– LL_HRTIM_EVENT_3</li> <li>– LL_HRTIM_EVENT_4</li> <li>– LL_HRTIM_EVENT_5</li> <li>– LL_HRTIM_EVENT_6</li> <li>– LL_HRTIM_EVENT_7</li> <li>– LL_HRTIM_EVENT_8</li> <li>– LL_HRTIM_EVENT_9</li> <li>– LL_HRTIM_EVENT_10</li> </ul> </li> <li>• <b>Polarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_EE_POLARITY_HIGH</li> <li>– LL_HRTIM_EE_POLARITY_LOW</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the timer counter is enabled.</li> <li>• Event polarity is only significant when event detection is level-sensitive.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EECR1 EE1POL LL_HRTIM_EE_SetPolarity</li> <li>• EECR1 EE2POL LL_HRTIM_EE_SetPolarity</li> <li>• EECR1 EE3POL LL_HRTIM_EE_SetPolarity</li> <li>• EECR1 EE4POL LL_HRTIM_EE_SetPolarity</li> <li>• EECR1 EE5POL LL_HRTIM_EE_SetPolarity</li> <li>• EECR2 EE6POL LL_HRTIM_EE_SetPolarity</li> <li>• EECR2 EE7POL LL_HRTIM_EE_SetPolarity</li> <li>• EECR2 EE8POL LL_HRTIM_EE_SetPolarity</li> <li>• EECR2 EE9POL LL_HRTIM_EE_SetPolarity</li> <li>• EECR2 EE10POL LL_HRTIM_EE_SetPolarity</li> </ul>                                                                                                                                                                          |

## LL\_HRTIM\_EE\_GetPolarity

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_EE_GetPolarity(<br/>HRTIM_TypeDef * HRTIMx, uint32_t Event)</code>                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description | Get actual polarity setting of an external event.                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_EVENT_1</li> <li>– LL_HRTIM_EVENT_2</li> <li>– LL_HRTIM_EVENT_3</li> <li>– LL_HRTIM_EVENT_4</li> <li>– LL_HRTIM_EVENT_5</li> <li>– LL_HRTIM_EVENT_6</li> <li>– LL_HRTIM_EVENT_7</li> <li>– LL_HRTIM_EVENT_8</li> <li>– LL_HRTIM_EVENT_9</li> <li>– LL_HRTIM_EVENT_10</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Polarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_EE_POLARITY_HIGH</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                           |

Reference Manual to  
LL API cross  
reference:

- LL\_HRTIM\_EE\_POLARITY\_LOW
- EECR1 EE1POL LL\_HRTIM\_EE\_GetPolarity
- EECR1 EE2POL LL\_HRTIM\_EE\_GetPolarity
- EECR1 EE3POL LL\_HRTIM\_EE\_GetPolarity
- EECR1 EE4POL LL\_HRTIM\_EE\_GetPolarity
- EECR1 EE5POL LL\_HRTIM\_EE\_GetPolarity
- EECR2 EE6POL LL\_HRTIM\_EE\_GetPolarity
- EECR2 EE7POL LL\_HRTIM\_EE\_GetPolarity
- EECR2 EE8POL LL\_HRTIM\_EE\_GetPolarity
- EECR2 EE9POL LL\_HRTIM\_EE\_GetPolarity
- EECR2 EE10POL LL\_HRTIM\_EE\_GetPolarity

### **LL\_HRTIM\_EE\_SetSensitivity**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_EE_SetSensitivity(HRTIM_TypeDef * HRTIMx, uint32_t Event, uint32_t Sensitivity)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Set the sensitivity of an external event.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EVENT_1</li> <li>- LL_HRTIM_EVENT_2</li> <li>- LL_HRTIM_EVENT_3</li> <li>- LL_HRTIM_EVENT_4</li> <li>- LL_HRTIM_EVENT_5</li> <li>- LL_HRTIM_EVENT_6</li> <li>- LL_HRTIM_EVENT_7</li> <li>- LL_HRTIM_EVENT_8</li> <li>- LL_HRTIM_EVENT_9</li> <li>- LL_HRTIM_EVENT_10</li> </ul> </li> <li>• <b>Sensitivity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EE_SENSITIVITY_LEVEL</li> <li>- LL_HRTIM_EE_SENSITIVITY_RISINGEDGE</li> <li>- LL_HRTIM_EE_SENSITIVITY_FALLINGEDGE</li> <li>- LL_HRTIM_EE_SENSITIVITY_BOTHEDGES</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EECR1 EE1SNS LL_HRTIM_EE_SetSensitivity</li> <li>• EECR1 EE2SNS LL_HRTIM_EE_SetSensitivity</li> <li>• EECR1 EE3SNS LL_HRTIM_EE_SetSensitivity</li> <li>• EECR1 EE4SNS LL_HRTIM_EE_SetSensitivity</li> <li>• EECR1 EE5SNS LL_HRTIM_EE_SetSensitivity</li> <li>• EECR2 EE6SNS LL_HRTIM_EE_SetSensitivity</li> <li>• EECR2 EE7SNS LL_HRTIM_EE_SetSensitivity</li> <li>• EECR2 EE8SNS LL_HRTIM_EE_SetSensitivity</li> <li>• EECR2 EE9SNS LL_HRTIM_EE_SetSensitivity</li> <li>• EECR2 EE10SNS LL_HRTIM_EE_SetSensitivity</li> </ul>                                                                                                                                                                                                                                                         |

**LL\_HRTIM\_EE\_GetSensitivity**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_EE_GetSensitivity(HRTIM_TypeDef * HRTIMx, uint32_t Event)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Get actual sensitivity setting of an external event.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EVENT_1</li> <li>- LL_HRTIM_EVENT_2</li> <li>- LL_HRTIM_EVENT_3</li> <li>- LL_HRTIM_EVENT_4</li> <li>- LL_HRTIM_EVENT_5</li> <li>- LL_HRTIM_EVENT_6</li> <li>- LL_HRTIM_EVENT_7</li> <li>- LL_HRTIM_EVENT_8</li> <li>- LL_HRTIM_EVENT_9</li> <li>- LL_HRTIM_EVENT_10</li> </ul> </li> </ul>                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Polarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EE_SENSITIVITY_LEVEL</li> <li>- LL_HRTIM_EE_SENSITIVITY_RISINGEDGE</li> <li>- LL_HRTIM_EE_SENSITIVITY_FALLINGEDGE</li> <li>- LL_HRTIM_EE_SENSITIVITY_BOTHEDGES</li> </ul> </li> </ul>                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EECR1 EE1SNS LL_HRTIM_EE_GetSensitivity</li> <li>• EECR1 EE2SNS LL_HRTIM_EE_GetSensitivity</li> <li>• EECR1 EE3SNS LL_HRTIM_EE_GetSensitivity</li> <li>• EECR1 EE4SNS LL_HRTIM_EE_GetSensitivity</li> <li>• EECR1 EE5SNS LL_HRTIM_EE_GetSensitivity</li> <li>• EECR2 EE6SNS LL_HRTIM_EE_GetSensitivity</li> <li>• EECR2 EE7SNS LL_HRTIM_EE_GetSensitivity</li> <li>• EECR2 EE8SNS LL_HRTIM_EE_GetSensitivity</li> <li>• EECR2 EE9SNS LL_HRTIM_EE_GetSensitivity</li> <li>• EECR2 EE10SNS LL_HRTIM_EE_GetSensitivity</li> </ul> |

**LL\_HRTIM\_EE\_SetFastMode**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_EE_SetFastMode(HRTIM_TypeDef * HRTIMx, uint32_t Event, uint32_t FastMode)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Set the fast mode of an external event.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EVENT_1</li> <li>- LL_HRTIM_EVENT_2</li> <li>- LL_HRTIM_EVENT_3</li> <li>- LL_HRTIM_EVENT_4</li> <li>- LL_HRTIM_EVENT_5</li> </ul> </li> <li>• <b>FastMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_EE_FASTMODE_DISABLE</li> <li>- LL_HRTIM_EE_FASTMODE_ENABLE</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>This function must not be called when the timer counter is enabled.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ECCR1 EE1FAST LL_HRTIM_EE_SetFastMode</li> <li>ECCR1 EE2FAST LL_HRTIM_EE_SetFastMode</li> <li>ECCR1 EE3FAST LL_HRTIM_EE_SetFastMode</li> <li>ECCR1 EE4FAST LL_HRTIM_EE_SetFastMode</li> <li>ECCR1 EE5FAST LL_HRTIM_EE_SetFastMode</li> <li>ECCR2 EE6FAST LL_HRTIM_EE_SetFastMode</li> <li>ECCR2 EE7FAST LL_HRTIM_EE_SetFastMode</li> <li>ECCR2 EE8FAST LL_HRTIM_EE_SetFastMode</li> <li>ECCR2 EE9FAST LL_HRTIM_EE_SetFastMode</li> <li>ECCR2 EE10FAST LL_HRTIM_EE_SetFastMode</li> </ul> |

### LL\_HRTIM\_EE\_GetFastMode

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_EE_GetFastMode(<br/>HRTIM_TypeDef * HRTIMx, uint32_t Event)</code>                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                              | Get actual fast mode setting of an external event.                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_EVENT_1</li> <li>– LL_HRTIM_EVENT_2</li> <li>– LL_HRTIM_EVENT_3</li> <li>– LL_HRTIM_EVENT_4</li> <li>– LL_HRTIM_EVENT_5</li> </ul> </li> </ul>                                                                                                                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li><b>FastMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_EE_FASTMODE_DISABLE</li> <li>– LL_HRTIM_EE_FASTMODE_ENABLE</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ECCR1 EE1FAST LL_HRTIM_EE_GetFastMode</li> <li>ECCR1 EE2FAST LL_HRTIM_EE_GetFastMode</li> <li>ECCR1 EE3FAST LL_HRTIM_EE_GetFastMode</li> <li>ECCR1 EE4FAST LL_HRTIM_EE_GetFastMode</li> <li>ECCR1 EE5FAST LL_HRTIM_EE_GetFastMode</li> <li>ECCR2 EE6FAST LL_HRTIM_EE_GetFastMode</li> <li>ECCR2 EE7FAST LL_HRTIM_EE_GetFastMode</li> <li>ECCR2 EE8FAST LL_HRTIM_EE_GetFastMode</li> <li>ECCR2 EE9FAST LL_HRTIM_EE_GetFastMode</li> <li>ECCR2 EE10FAST LL_HRTIM_EE_GetFastMode</li> </ul> |

### LL\_HRTIM\_EE\_SetFilter

|                      |                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_EE_SetFilter(<br/>HRTIM_TypeDef * HRTIMx, uint32_t Event, uint32_t Filter)</code>                                                                                                                           |
| Function description | Set the digital noise filter of a external event.                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Event:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_EVENT_6</li> </ul> </li> </ul> |

- LL\_HRTIM\_EVENT\_7
- LL\_HRTIM\_EVENT\_8
- LL\_HRTIM\_EVENT\_9
- LL\_HRTIM\_EVENT\_10
- **Filter:** This parameter can be one of the following values:
  - LL\_HRTIM\_EE\_FILTER\_NONE
  - LL\_HRTIM\_EE\_FILTER\_1
  - LL\_HRTIM\_EE\_FILTER\_2
  - LL\_HRTIM\_EE\_FILTER\_3
  - LL\_HRTIM\_EE\_FILTER\_4
  - LL\_HRTIM\_EE\_FILTER\_5
  - LL\_HRTIM\_EE\_FILTER\_6
  - LL\_HRTIM\_EE\_FILTER\_7
  - LL\_HRTIM\_EE\_FILTER\_8
  - LL\_HRTIM\_EE\_FILTER\_9
  - LL\_HRTIM\_EE\_FILTER\_10
  - LL\_HRTIM\_EE\_FILTER\_11
  - LL\_HRTIM\_EE\_FILTER\_12
  - LL\_HRTIM\_EE\_FILTER\_13
  - LL\_HRTIM\_EE\_FILTER\_14
  - LL\_HRTIM\_EE\_FILTER\_15

## Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- EECR3 EE6F LL\_HRTIM\_EE\_SetFilter
- EECR3 EE7F LL\_HRTIM\_EE\_SetFilter
- EECR3 EE8F LL\_HRTIM\_EE\_SetFilter
- EECR3 EE9F LL\_HRTIM\_EE\_SetFilter
- EECR3 EE10F LL\_HRTIM\_EE\_SetFilter

**LL\_HRTIM\_EE\_GetFilter**

## Function name

```
STATIC_INLINE uint32_t LL_HRTIM_EE_GetFilter
(HRTIM_TypeDef * HRTIMx, uint32_t Event)
```

## Function description

Get actual digital noise filter setting of a external event.

## Parameters

- **HRTIMx:** High Resolution Timer instance
- **Event:** This parameter can be one of the following values:
  - LL\_HRTIM\_EVENT\_6
  - LL\_HRTIM\_EVENT\_7
  - LL\_HRTIM\_EVENT\_8
  - LL\_HRTIM\_EVENT\_9
  - LL\_HRTIM\_EVENT\_10

## Return values

- **Filter:** This parameter can be one of the following values:
  - LL\_HRTIM\_EE\_FILTER\_NONE
  - LL\_HRTIM\_EE\_FILTER\_1
  - LL\_HRTIM\_EE\_FILTER\_2
  - LL\_HRTIM\_EE\_FILTER\_3
  - LL\_HRTIM\_EE\_FILTER\_4
  - LL\_HRTIM\_EE\_FILTER\_5
  - LL\_HRTIM\_EE\_FILTER\_6
  - LL\_HRTIM\_EE\_FILTER\_7
  - LL\_HRTIM\_EE\_FILTER\_8

- LL\_HRTIM\_EE\_FILTER\_9
- LL\_HRTIM\_EE\_FILTER\_10
- LL\_HRTIM\_EE\_FILTER\_11
- LL\_HRTIM\_EE\_FILTER\_12
- LL\_HRTIM\_EE\_FILTER\_13
- LL\_HRTIM\_EE\_FILTER\_14
- LL\_HRTIM\_EE\_FILTER\_15

Reference Manual to  
LL API cross  
reference:

- EECR3 EE6F LL\_HRTIM\_EE\_GetFilter
- EECR3 EE7F LL\_HRTIM\_EE\_GetFilter
- EECR3 EE8F LL\_HRTIM\_EE\_GetFilter
- EECR3 EE9F LL\_HRTIM\_EE\_GetFilter
- EECR3 EE10F LL\_HRTIM\_EE\_GetFilter

### **LL\_HRTIM\_EE\_SetPrescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_EE_SetPrescaler(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Prescaler)</code>                                                                                                                                                                                                                                                                               |
| Function description                              | Set the external event prescaler.                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Prescaler:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_EE_PRESCALER_DIV1</li> <li>- LL_HRTIM_EE_PRESCALER_DIV2</li> <li>- LL_HRTIM_EE_PRESCALER_DIV4</li> <li>- LL_HRTIM_EE_PRESCALER_DIV8</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EECR3 EEVSD LL_HRTIM_EE_SetPrescaler</li> </ul>                                                                                                                                                                                                                                                                                                      |

### **LL\_HRTIM\_EE\_GetPrescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_EE_GetPrescaler(<br/>    HRTIM_TypeDef * HRTIMx)</code>                                                                                                                                                                                                                                      |
| Function description                              | Get actual external event prescaler setting.                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Prescaler:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_EE_PRESCALER_DIV1</li> <li>- LL_HRTIM_EE_PRESCALER_DIV2</li> <li>- LL_HRTIM_EE_PRESCALER_DIV4</li> <li>- LL_HRTIM_EE_PRESCALER_DIV8</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• EECR3 EEVSD LL_HRTIM_EE_GetPrescaler</li> </ul>                                                                                                                                                                                                                                             |

**LL\_HRTIM\_FLT\_Config**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_FLT_Config<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Fault, uint32_t<br/>Configuration)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description                              | Configure fault signal conditioning.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIMFAULT_1</li> <li>– LL_HRTIMFAULT_2</li> <li>– LL_HRTIMFAULT_3</li> <li>– LL_HRTIMFAULT_4</li> <li>– LL_HRTIMFAULT_5</li> </ul> </li> <li>• <b>Configuration:</b> This parameter must be a combination of all the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_FLT_SRC_DIGITALINPUT or LL_HRTIM_FLT_SRC_INTERNAL</li> <li>– LL_HRTIM_FLT_POLARITY_LOW or LL_HRTIM_FLT_POLARITY_HIGH</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the fault channel is enabled.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• FLTINR1 FLT1P LL_HRTIM_FLT_Config</li> <li>• FLTINR1 FLT1SRC LL_HRTIM_FLT_Config</li> <li>• FLTINR1 FLT2P LL_HRTIM_FLT_Config</li> <li>• FLTINR1 FLT2SRC LL_HRTIM_FLT_Config</li> <li>• FLTINR1 FLT3P LL_HRTIM_FLT_Config</li> <li>• FLTINR1 FLT3SRC LL_HRTIM_FLT_Config</li> <li>• FLTINR1 FLT4P LL_HRTIM_FLT_Config</li> <li>• FLTINR1 FLT4SRC LL_HRTIM_FLT_Config</li> <li>• FLTINR2 FLT5P LL_HRTIM_FLT_Config</li> <li>• FLTINR2 FLT5SRC LL_HRTIM_FLT_Config</li> </ul>                                                                                                                                 |

**LL\_HRTIM\_FLT\_SetSrc**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_FLT_SetSrc<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Fault, uint32_t Src)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Set the source of a fault signal.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIMFAULT_1</li> <li>– LL_HRTIMFAULT_2</li> <li>– LL_HRTIMFAULT_3</li> <li>– LL_HRTIMFAULT_4</li> <li>– LL_HRTIMFAULT_5</li> </ul> </li> <li>• <b>Src:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_FLT_SRC_DIGITALINPUT</li> <li>– LL_HRTIM_FLT_SRC_INTERNAL</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

|                                             |                                                                                                                                                                                                                                                                           |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>This function must not be called when the fault channel is enabled.</li> </ul>                                                                                                                                                     |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>FLTINR1 FLT1SRC LL_HRTIM_FLT_SetSrc</li> <li>FLTINR1 FLT2SRC LL_HRTIM_FLT_SetSrc</li> <li>FLTINR1 FLT3SRC LL_HRTIM_FLT_SetSrc</li> <li>FLTINR1 FLT4SRC LL_HRTIM_FLT_SetSrc</li> <li>FLTINR2 FLT5SRC LL_HRTIM_FLT_SetSrc</li> </ul> |

### LL\_HRTIM\_FLT\_GetSrc

|                                             |                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_HRTIM_FLT_GetSrc(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Fault)</code>                                                                                                                                                                                                                                            |
| Function description                        | Get actual source of a fault signal.                                                                                                                                                                                                                                                                                                                  |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_HRTIM_FAULT_1</li> <li>LL_HRTIM_FAULT_2</li> <li>LL_HRTIM_FAULT_3</li> <li>LL_HRTIM_FAULT_4</li> <li>LL_HRTIM_FAULT_5</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>Src:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_HRTIM_FLT_SRC_DIGITALINPUT</li> <li>LL_HRTIM_FLT_SRC_INTERNAL</li> </ul> </li> </ul>                                                                                                                  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>FLTINR1 FLT1SRC LL_HRTIM_FLT_GetSrc</li> <li>FLTINR1 FLT2SRC LL_HRTIM_FLT_GetSrc</li> <li>FLTINR1 FLT3SRC LL_HRTIM_FLT_GetSrc</li> <li>FLTINR1 FLT4SRC LL_HRTIM_FLT_GetSrc</li> <li>FLTINR2 FLT5SRC LL_HRTIM_FLT_GetSrc</li> </ul>                                                                             |

### LL\_HRTIM\_FLT\_SetPolarity

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_FLT_SetPolarity(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Fault, uint32_t Polarity)</code>                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description | Set the polarity of a fault signal.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_HRTIM_FAULT_1</li> <li>LL_HRTIM_FAULT_2</li> <li>LL_HRTIM_FAULT_3</li> <li>LL_HRTIM_FAULT_4</li> <li>LL_HRTIM_FAULT_5</li> </ul> </li> <li><b>Polarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_HRTIM_FLT_POLARITY_LOW</li> <li>LL_HRTIM_FLT_POLARITY_HIGH</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>This function must not be called when the fault channel is enabled.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Reference Manual to  | <ul style="list-style-type: none"> <li>FLTINR1 FLT1P LL_HRTIM_FLT_SetPolarity</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

- LL API cross reference:
- `FLTINR1 FLT2P LL_HRTIM_FLT_SetPolarity`
  - `FLTINR1 FLT3P LL_HRTIM_FLT_SetPolarity`
  - `FLTINR1 FLT4P LL_HRTIM_FLT_SetPolarity`
  - `FLTINR2 FLT5P LL_HRTIM_FLT_SetPolarity`

### **LL\_HRTIM\_FLT\_GetPolarity**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_HRTIM_FLT_GetPolarity(HRTIM_TypeDef * HRTIMx, uint32_t Fault)</code>                                                                                                                                                                                                                                                                                                                          |
| Function description                        | Get actual polarity of a fault signal.                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_HRTIMFAULT_1</code></li> <li>- <code>LL_HRTIMFAULT_2</code></li> <li>- <code>LL_HRTIMFAULT_3</code></li> <li>- <code>LL_HRTIMFAULT_4</code></li> <li>- <code>LL_HRTIMFAULT_5</code></li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Polarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_HRTIM_FLT_POLARITY_LOW</code></li> <li>- <code>LL_HRTIM_FLT_POLARITY_HIGH</code></li> </ul> </li> </ul>                                                                                                                                                          |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• <code>FLTINR1 FLT1P LL_HRTIM_FLT_GetPolarity</code></li> <li>• <code>FLTINR1 FLT2P LL_HRTIM_FLT_GetPolarity</code></li> <li>• <code>FLTINR1 FLT3P LL_HRTIM_FLT_GetPolarity</code></li> <li>• <code>FLTINR1 FLT4P LL_HRTIM_FLT_GetPolarity</code></li> <li>• <code>FLTINR2 FLT5P LL_HRTIM_FLT_GetPolarity</code></li> </ul>                                                             |

### **LL\_HRTIM\_FLT\_SetFilter**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_FLT_SetFilter(HRTIM_TypeDef * HRTIMx, uint32_t Fault, uint32_t Filter)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description | Set the digital noise filter of a fault signal.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_HRTIMFAULT_1</code></li> <li>- <code>LL_HRTIMFAULT_2</code></li> <li>- <code>LL_HRTIMFAULT_3</code></li> <li>- <code>LL_HRTIMFAULT_4</code></li> <li>- <code>LL_HRTIMFAULT_5</code></li> </ul> </li> <li>• <b>Filter:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_HRTIM_FLT_FILTER_NONE</code></li> <li>- <code>LL_HRTIM_FLT_FILTER_1</code></li> <li>- <code>LL_HRTIM_FLT_FILTER_2</code></li> <li>- <code>LL_HRTIM_FLT_FILTER_3</code></li> <li>- <code>LL_HRTIM_FLT_FILTER_4</code></li> <li>- <code>LL_HRTIM_FLT_FILTER_5</code></li> <li>- <code>LL_HRTIM_FLT_FILTER_6</code></li> <li>- <code>LL_HRTIM_FLT_FILTER_7</code></li> <li>- <code>LL_HRTIM_FLT_FILTER_8</code></li> <li>- <code>LL_HRTIM_FLT_FILTER_9</code></li> <li>- <code>LL_HRTIM_FLT_FILTER_10</code></li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_FLT_FILTER_11</li> <li>- LL_HRTIM_FLT_FILTER_12</li> <li>- LL_HRTIM_FLT_FILTER_13</li> <li>- LL_HRTIM_FLT_FILTER_14</li> <li>- LL_HRTIM_FLT_FILTER_15</li> </ul>                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function must not be called when the fault channel is enabled.</li> </ul>                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• FLTINR1 FLT1F LL_HRTIM_FLT_SetFilter</li> <li>• FLTINR1 FLT2F LL_HRTIM_FLT_SetFilter</li> <li>• FLTINR1 FLT3F LL_HRTIM_FLT_SetFilter</li> <li>• FLTINR1 FLT4F LL_HRTIM_FLT_SetFilter</li> <li>• FLTINR2 FLT5F LL_HRTIM_FLT_SetFilter</li> </ul> |

### LL\_HRTIM\_FLT\_GetFilter

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>STATIC_INLINE uint32_t LL_HRTIM_FLT_GetFilter(<br/>HRTIM_TypeDef * HRTIMx, uint32_t Fault)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                              | Get actual digital noise filter setting of a fault signal.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_FAULT_1</li> <li>- LL_HRTIM_FAULT_2</li> <li>- LL_HRTIM_FAULT_3</li> <li>- LL_HRTIM_FAULT_4</li> <li>- LL_HRTIM_FAULT_5</li> </ul> </li> <li>• <b>Filter:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_FLT_FILTER_NONE</li> <li>- LL_HRTIM_FLT_FILTER_1</li> <li>- LL_HRTIM_FLT_FILTER_2</li> <li>- LL_HRTIM_FLT_FILTER_3</li> <li>- LL_HRTIM_FLT_FILTER_4</li> <li>- LL_HRTIM_FLT_FILTER_5</li> <li>- LL_HRTIM_FLT_FILTER_6</li> <li>- LL_HRTIM_FLT_FILTER_7</li> <li>- LL_HRTIM_FLT_FILTER_8</li> <li>- LL_HRTIM_FLT_FILTER_9</li> <li>- LL_HRTIM_FLT_FILTER_10</li> <li>- LL_HRTIM_FLT_FILTER_11</li> <li>- LL_HRTIM_FLT_FILTER_12</li> <li>- LL_HRTIM_FLT_FILTER_13</li> <li>- LL_HRTIM_FLT_FILTER_14</li> <li>- LL_HRTIM_FLT_FILTER_15</li> </ul> </li> </ul> |
| Return values                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• FLTINR1 FLT1F LL_HRTIM_FLT_GetFilter</li> <li>• FLTINR1 FLT2F LL_HRTIM_FLT_GetFilter</li> <li>• FLTINR1 FLT3F LL_HRTIM_FLT_GetFilter</li> <li>• FLTINR1 FLT4F LL_HRTIM_FLT_GetFilter</li> <li>• FLTINR2 FLT5F LL_HRTIM_FLT_GetFilter</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**LL\_HRTIM\_FLT\_SetPrescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_FLT_SetPrescaler(HRTIM_TypeDef * HRTIMx, uint32_t Prescaler)</code></b>                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Set the fault circuitry prescaler.                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Prescaler:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_HRTIM_FLT_PRESCALER_DIV1</code></li> <li>– <code>LL_HRTIM_FLT_PRESCALER_DIV2</code></li> <li>– <code>LL_HRTIM_FLT_PRESCALER_DIV4</code></li> <li>– <code>LL_HRTIM_FLT_PRESCALER_DIV8</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>FLTINR2.FLTSD</code> <code>LL_HRTIM_FLT_SetPrescaler</code></li> </ul>                                                                                                                                                                                                                                                                                                                       |

**LL\_HRTIM\_FLT\_GetPrescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_FLT_GetPrescaler(HRTIM_TypeDef * HRTIMx)</code></b>                                                                                                                                                                                                                                                                                      |
| Function description                              | Get actual fault circuitry prescaler setting.                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Prescaler:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_HRTIM_FLT_PRESCALER_DIV1</code></li> <li>– <code>LL_HRTIM_FLT_PRESCALER_DIV2</code></li> <li>– <code>LL_HRTIM_FLT_PRESCALER_DIV4</code></li> <li>– <code>LL_HRTIM_FLT_PRESCALER_DIV8</code></li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>FLTINR2.FLTSD</code> <code>LL_HRTIM_FLT_GetPrescaler</code></li> </ul>                                                                                                                                                                                                                                                              |

**LL\_HRTIM\_FLT\_Lock**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_FLT_Lock(HRTIM_TypeDef * HRTIMx, uint32_t Fault)</code></b>                                                                                                                                                                                                                                                                                                                                    |
| Function description                              | Lock the fault signal conditioning settings.                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_HRTIM_FAULT_1</code></li> <li>– <code>LL_HRTIM_FAULT_2</code></li> <li>– <code>LL_HRTIM_FAULT_3</code></li> <li>– <code>LL_HRTIM_FAULT_4</code></li> <li>– <code>LL_HRTIM_FAULT_5</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>FLTINR1.FLT1LCK</code> <code>LL_HRTIM_FLT_Lock</code></li> <li>• <code>FLTINR1.FLT2LCK</code> <code>LL_HRTIM_FLT_Lock</code></li> </ul>                                                                                                                                                                                                                                               |

- reference:
- `FLTINR1 FLT3LCK LL_HRTIM_FLT_Lock`
  - `FLTINR1 FLT4LCK LL_HRTIM_FLT_Lock`
  - `FLTINR2 FLT5LCK LL_HRTIM_FLT_Lock`

### **LL\_HRTIM\_FLT\_Enable**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_FLT_Enable<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Fault)</code>                                                                                                                                                                                                                                                                                                                              |
| Function description                              | Enable the fault circuitry for the designated fault input.                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_HRTIMFAULT_1</code></li> <li>- <code>LL_HRTIMFAULT_2</code></li> <li>- <code>LL_HRTIMFAULT_3</code></li> <li>- <code>LL_HRTIMFAULT_4</code></li> <li>- <code>LL_HRTIMFAULT_5</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>FLTINR1 FLT1E LL_HRTIM_FLT_Enable</code></li> <li>• <code>FLTINR1 FLT2E LL_HRTIM_FLT_Enable</code></li> <li>• <code>FLTINR1 FLT3E LL_HRTIM_FLT_Enable</code></li> <li>• <code>FLTINR1 FLT4E LL_HRTIM_FLT_Enable</code></li> <li>• <code>FLTINR2 FLT5E LL_HRTIM_FLT_Enable</code></li> </ul>                                                                                      |

### **LL\_HRTIM\_FLT\_Disable**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_FLT_Disable<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Fault)</code>                                                                                                                                                                                                                                                                                                                             |
| Function description                              | Disable the fault circuitry for the designated fault input.                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Fault:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_HRTIMFAULT_1</code></li> <li>- <code>LL_HRTIMFAULT_2</code></li> <li>- <code>LL_HRTIMFAULT_3</code></li> <li>- <code>LL_HRTIMFAULT_4</code></li> <li>- <code>LL_HRTIMFAULT_5</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>FLTINR1 FLT1E LL_HRTIM_FLT_Disable</code></li> <li>• <code>FLTINR1 FLT2E LL_HRTIM_FLT_Disable</code></li> <li>• <code>FLTINR1 FLT3E LL_HRTIM_FLT_Disable</code></li> <li>• <code>FLTINR1 FLT4E LL_HRTIM_FLT_Disable</code></li> <li>• <code>FLTINR2 FLT5E LL_HRTIM_FLT_Disable</code></li> </ul>                                                                                 |

### **LL\_HRTIM\_FLT\_IsEnabled**

|                      |                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_FLT_IsEnabled<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Fault)</code> |
| Function description | Indicate whether the fault circuitry is enabled for a given fault input.                                  |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance *</li> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Fault:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_FAULT_1</li> <li>- LL_HRTIM_FAULT_2</li> <li>- LL_HRTIM_FAULT_3</li> <li>- LL_HRTIM_FAULT_4</li> <li>- LL_HRTIM_FAULT_5</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of FLTEN bit in HRTIM_FLTINRx register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>FLTINR1 FLT1E LL_HRTIM_FLT_IsEnabled</li> <li>FLTINR1 FLT2E LL_HRTIM_FLT_IsEnabled</li> <li>FLTINR1 FLT3E LL_HRTIM_FLT_IsEnabled</li> <li>FLTINR1 FLT4E LL_HRTIM_FLT_IsEnabled</li> <li>FLTINR2 FLT5E LL_HRTIM_FLT_IsEnabled</li> </ul>                                                                                                                                                     |

### LL\_HRTIM\_BM\_Config

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_HRTIM_BM_Config(<br/>  HRTIM_TypeDef * HRTIMx, uint32_t Configuration)</code>                                                                                                                                                                                                                                                                                                                                                                     |
| Function description                              | Configure the burst mode controller.                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Configuration:</b> This parameter must be a combination of all the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_BM_MODE_SINGLESHOT or LL_HRTIM_BM_MODE_CONTINOUS</li> <li>- LL_HRTIM_BM_CLKSRC_MASTER or ... or LL_HRTIM_BM_CLKSRC_FHRTIM</li> <li>- LL_HRTIM_BM_PRESCALER_DIV1 or ... LL_HRTIM_BM_PRESCALER_DIV32768</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>BMCR BMOM LL_HRTIM_BM_Config</li> <li>BMCR BMCLK LL_HRTIM_BM_Config</li> <li>BMCR BMPRSC LL_HRTIM_BM_Config</li> </ul>                                                                                                                                                                                                                                                                                                                  |

### LL\_HRTIM\_BM\_SetMode

|                                                   |                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_HRTIM_BM_SetMode(<br/>  HRTIM_TypeDef * HRTIMx, uint32_t Mode)</code>                                                                                                                                                                                                        |
| Function description                              | Set the burst mode controller operating mode.                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Mode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_BM_MODE_SINGLESHOT</li> <li>- LL_HRTIM_BM_MODE_CONTINOUS</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>BMCR BMOM LL_HRTIM_BM_SetMode</li> </ul>                                                                                                                                                                                                                           |

**LL\_HRTIM\_BM\_GetMode**

|                                                   |                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_HRTIM_BM_GetMode<br/>(HRTIM_TypeDef * HRTIMx)</code>                                                                                                                                                                |
| Function description                              | Get actual burst mode controller operating mode.                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Mode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_BM_MODE_SINGLESHOT</li> <li>– LL_HRTIM_BM_MODE_CONTINOUS</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BMCR BMOM LL_HRTIM_BM_GetMode</li> </ul>                                                                                                                                                                    |

**LL\_HRTIM\_BM\_SetClockSrc**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_HRTIM_BM_SetClockSrc<br/>(HRTIM_TypeDef * HRTIMx, uint32_t ClockSrc)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Set the burst mode controller clock source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>ClockSrc:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_BM_CLKSRC_MASTER</li> <li>– LL_HRTIM_BM_CLKSRC_TIMER_A</li> <li>– LL_HRTIM_BM_CLKSRC_TIMER_B</li> <li>– LL_HRTIM_BM_CLKSRC_TIMER_C</li> <li>– LL_HRTIM_BM_CLKSRC_TIMER_D</li> <li>– LL_HRTIM_BM_CLKSRC_TIMER_E</li> <li>– LL_HRTIM_BM_CLKSRC_TIM16_OC</li> <li>– LL_HRTIM_BM_CLKSRC_TIM17_OC</li> <li>– LL_HRTIM_BM_CLKSRC_TIM7_TRGO</li> <li>– LL_HRTIM_BM_CLKSRC_FHRTIM</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BMCR BMCLK LL_HRTIM_BM_SetClockSrc</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

**LL\_HRTIM\_BM\_GetClockSrc**

|                      |                                                                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_HRTIM_BM_GetClockSrc<br/>(HRTIM_TypeDef * HRTIMx)</code>                                                                                                                                                                                                                                          |
| Function description | Get actual burst mode controller clock source.                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                                                                                                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>ClockSrc:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_BM_CLKSRC_MASTER</li> <li>– LL_HRTIM_BM_CLKSRC_TIMER_A</li> <li>– LL_HRTIM_BM_CLKSRC_TIMER_B</li> <li>– LL_HRTIM_BM_CLKSRC_TIMER_C</li> </ul> </li> </ul> |

- LL\_HRTIM\_BM\_CLKSRC\_TIMER\_D
- LL\_HRTIM\_BM\_CLKSRC\_TIMER\_E
- LL\_HRTIM\_BM\_CLKSRC\_TIM16\_OC
- LL\_HRTIM\_BM\_CLKSRC\_TIM17\_OC
- LL\_HRTIM\_BM\_CLKSRC\_TIM7\_TRGO
- LL\_HRTIM\_BM\_CLKSRC\_FHRTIM

Reference Manual to  
LL API cross  
reference:

- BMCR BMCLK LL\_HRTIM\_BM\_GetClockSrc

### **LL\_HRTIM\_BM\_SetPrescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_BM_SetPrescaler(<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Prescaler)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description                              | Set the burst mode controller prescaler.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Prescaler:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_BM_PRESCALER_DIV1</li> <li>- LL_HRTIM_BM_PRESCALER_DIV2</li> <li>- LL_HRTIM_BM_PRESCALER_DIV4</li> <li>- LL_HRTIM_BM_PRESCALER_DIV8</li> <li>- LL_HRTIM_BM_PRESCALER_DIV16</li> <li>- LL_HRTIM_BM_PRESCALER_DIV32</li> <li>- LL_HRTIM_BM_PRESCALER_DIV64</li> <li>- LL_HRTIM_BM_PRESCALER_DIV128</li> <li>- LL_HRTIM_BM_PRESCALER_DIV256</li> <li>- LL_HRTIM_BM_PRESCALER_DIV512</li> <li>- LL_HRTIM_BM_PRESCALER_DIV1024</li> <li>- LL_HRTIM_BM_PRESCALER_DIV2048</li> <li>- LL_HRTIM_BM_PRESCALER_DIV4096</li> <li>- LL_HRTIM_BM_PRESCALER_DIV8192</li> <li>- LL_HRTIM_BM_PRESCALER_DIV16384</li> <li>- LL_HRTIM_BM_PRESCALER_DIV32768</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BMCR BMPRSC LL_HRTIM_BM_SetPrescaler</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

### **LL\_HRTIM\_BM\_GetPrescaler**

|                      |                                                                                                                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_BM_GetPrescaler(<br/>(HRTIM_TypeDef * HRTIMx)</code></b>                                                                                                                                                       |
| Function description | Get actual burst mode controller prescaler setting.                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Prescaler:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_BM_PRESCALER_DIV1</li> <li>- LL_HRTIM_BM_PRESCALER_DIV2</li> </ul> </li> </ul> |

- LL\_HRTIM\_BM\_PRESCALER\_DIV4
- LL\_HRTIM\_BM\_PRESCALER\_DIV8
- LL\_HRTIM\_BM\_PRESCALER\_DIV16
- LL\_HRTIM\_BM\_PRESCALER\_DIV32
- LL\_HRTIM\_BM\_PRESCALER\_DIV64
- LL\_HRTIM\_BM\_PRESCALER\_DIV128
- LL\_HRTIM\_BM\_PRESCALER\_DIV256
- LL\_HRTIM\_BM\_PRESCALER\_DIV512
- LL\_HRTIM\_BM\_PRESCALER\_DIV1024
- LL\_HRTIM\_BM\_PRESCALER\_DIV2048
- LL\_HRTIM\_BM\_PRESCALER\_DIV4096
- LL\_HRTIM\_BM\_PRESCALER\_DIV8192
- LL\_HRTIM\_BM\_PRESCALER\_DIV16384
- LL\_HRTIM\_BM\_PRESCALER\_DIV32768

Reference Manual to  
LL API cross  
reference:

- BMCR BMPRSC LL\_HRTIM\_BM\_GetPrescaler

### **LL\_HRTIM\_BM\_EnablePreload**

Function name      **`_STATIC_INLINE void LL_HRTIM_BM_EnablePreload(HRTIM_TypeDef * HRTIMx)`**

Function description      Enable burst mode compare and period registers preload.

Parameters      • **HRTIMx:** High Resolution Timer instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:

- BMCR BMPREN LL\_HRTIM\_BM\_EnablePreload

### **LL\_HRTIM\_BM\_DisablePreload**

Function name      **`_STATIC_INLINE void LL_HRTIM_BM_DisablePreload(HRTIM_TypeDef * HRTIMx)`**

Function description      Disable burst mode compare and period registers preload.

Parameters      • **HRTIMx:** High Resolution Timer instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:

- BMCR BMPREN LL\_HRTIM\_BM\_DisablePreload

### **LL\_HRTIM\_BM\_IsEnabledPreload**

Function name      **`_STATIC_INLINE uint32_t LL_HRTIM_BM_IsEnabledPreload(HRTIM_TypeDef * HRTIMx)`**

Function description      Indicate whether burst mode compare and period registers are preloaded.

Parameters      • **HRTIMx:** High Resolution Timer instance

- |                                                                    |                                                                                                                                                                      |
|--------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values<br>Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <b>State:</b> of BMPREN bit in HRTIM_BMCR register (1 or 0).</li> <li>• BMCR BMPREN LL_HRTIM_BM_IsEnabledPreload</li> </ul> |
|--------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### **LL\_HRTIM\_BM\_SetTrig**

|                                                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|--------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name<br>Function description<br>Parameters                | <pre><b>_STATIC_INLINE void LL_HRTIM_BM_SetTrig</b><br/> <b>(HRTIM_TypeDef * HRTIMx, uint32_t Trig)</b></pre> <p>Set the burst mode controller trigger.</p> <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Trig:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>– LL_HRTIM_BM_TRIG_NONE</li> <li>– LL_HRTIM_BM_TRIG_MASTER_RESET</li> <li>– LL_HRTIM_BM_TRIG_MASTER_REPETITION</li> <li>– LL_HRTIM_BM_TRIG_MASTER_CMP1</li> <li>– LL_HRTIM_BM_TRIG_MASTER_CMP2</li> <li>– LL_HRTIM_BM_TRIG_MASTER_CMP3</li> <li>– LL_HRTIM_BM_TRIG_MASTER_CMP4</li> <li>– LL_HRTIM_BM_TRIG_TIMA_RESET</li> <li>– LL_HRTIM_BM_TRIG_TIMA_REPETITION</li> <li>– LL_HRTIM_BM_TRIG_TIMA_CMP1</li> <li>– LL_HRTIM_BM_TRIG_TIMA_CMP2</li> <li>– LL_HRTIM_BM_TRIG_TIMB_RESET</li> <li>– LL_HRTIM_BM_TRIG_TIMB_REPETITION</li> <li>– LL_HRTIM_BM_TRIG_TIMB_CMP1</li> <li>– LL_HRTIM_BM_TRIG_TIMB_CMP2</li> <li>– LL_HRTIM_BM_TRIG_TIMC_RESET</li> <li>– LL_HRTIM_BM_TRIG_TIMC_REPETITION</li> <li>– LL_HRTIM_BM_TRIG_TIMC_CMP1</li> <li>– LL_HRTIM_BM_TRIG_TIMC_CMP2</li> <li>– LL_HRTIM_BM_TRIG_TIMD_RESET</li> <li>– LL_HRTIM_BM_TRIG_TIMD_REPETITION</li> <li>– LL_HRTIM_BM_TRIG_TIMD_CMP1</li> <li>– LL_HRTIM_BM_TRIG_TIMD_CMP2</li> <li>– LL_HRTIM_BM_TRIG_TIME_RESET</li> <li>– LL_HRTIM_BM_TRIG_TIME_REPETITION</li> <li>– LL_HRTIM_BM_TRIG_TIME_CMP1</li> <li>– LL_HRTIM_BM_TRIG_TIME_CMP2</li> <li>– LL_HRTIM_BM_TRIG_TIMA_EVENT7</li> <li>– LL_HRTIM_BM_TRIG_TIMD_EVENT8</li> <li>– LL_HRTIM_BM_TRIG_EVENT_7</li> <li>– LL_HRTIM_BM_TRIG_EVENT_8</li> <li>– LL_HRTIM_BM_TRIG_EVENT_ONCHIP</li> </ul> </li> </ul> |
| Return values<br>Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <b>None</b></li> <li>• BMTRGR SW LL_HRTIM_BM_SetTrig</li> <li>• BMTRGR MSTRST LL_HRTIM_BM_SetTrig</li> <li>• BMTRGR MSTREP LL_HRTIM_BM_SetTrig</li> <li>• BMTRGR MSTCMP1 LL_HRTIM_BM_SetTrig</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|                                                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

- BMTRGR MSTCMP2 LL\_HRTIM\_BM\_SetTrig
- BMTRGR MSTCMP3 LL\_HRTIM\_BM\_SetTrig
- BMTRGR MSTCMP4 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TARST LL\_HRTIM\_BM\_SetTrig
- BMTRGR TAREP LL\_HRTIM\_BM\_SetTrig
- BMTRGR TACMP1 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TACMP2 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TBRST LL\_HRTIM\_BM\_SetTrig
- BMTRGR TBREP LL\_HRTIM\_BM\_SetTrig
- BMTRGR TBCMP1 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TBCMP2 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TCRST LL\_HRTIM\_BM\_SetTrig
- BMTRGR TCREP LL\_HRTIM\_BM\_SetTrig
- BMTRGR TCCMP1 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TCCMP2 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TDRST LL\_HRTIM\_BM\_SetTrig
- BMTRGR TDREP LL\_HRTIM\_BM\_SetTrig
- BMTRGR TDCCMP1 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TDCCMP2 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TERST LL\_HRTIM\_BM\_SetTrig
- BMTRGR TEREP LL\_HRTIM\_BM\_SetTrig
- BMTRGR TECMP1 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TECMP2 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TAAEV7 LL\_HRTIM\_BM\_SetTrig
- BMTRGR TAAEV8 LL\_HRTIM\_BM\_SetTrig
- BMTRGR EEV7 LL\_HRTIM\_BM\_SetTrig
- BMTRGR EEV8 LL\_HRTIM\_BM\_SetTrig
- BMTRGR OCHIPEV LL\_HRTIM\_BM\_SetTrig

### **LL\_HRTIM\_BM\_GetTrig**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_BM_GetTrig(<br/>                  <b>HRTIM_TypeDef</b> * HRTIMx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description | Get actual burst mode controller trigger.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Trig:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_BM_TRIG_NONE</li> <li>- LL_HRTIM_BM_TRIG_MASTER_RESET</li> <li>- LL_HRTIM_BM_TRIG_MASTER_REPETITION</li> <li>- LL_HRTIM_BM_TRIG_MASTER_CMP1</li> <li>- LL_HRTIM_BM_TRIG_MASTER_CMP2</li> <li>- LL_HRTIM_BM_TRIG_MASTER_CMP3</li> <li>- LL_HRTIM_BM_TRIG_MASTER_CMP4</li> <li>- LL_HRTIM_BM_TRIG_TIMA_RESET</li> <li>- LL_HRTIM_BM_TRIG_TIMA_REPETITION</li> <li>- LL_HRTIM_BM_TRIG_TIMA_CMP1</li> <li>- LL_HRTIM_BM_TRIG_TIMA_CMP2</li> <li>- LL_HRTIM_BM_TRIG_TIMB_RESET</li> <li>- LL_HRTIM_BM_TRIG_TIMB_REPETITION</li> <li>- LL_HRTIM_BM_TRIG_TIMB_CMP1</li> </ul> </li> </ul> |

- LL\_HRTIM\_BM\_TRIG\_TIMB\_CMP2
- LL\_HRTIM\_BM\_TRIG\_TIMC\_RESET
- LL\_HRTIM\_BM\_TRIG\_TIMC\_REPETITION
- LL\_HRTIM\_BM\_TRIG\_TIMC\_CMP1
- LL\_HRTIM\_BM\_TRIG\_TIMC\_CMP2
- LL\_HRTIM\_BM\_TRIG\_TIMD\_RESET
- LL\_HRTIM\_BM\_TRIG\_TIMD\_REPETITION
- LL\_HRTIM\_BM\_TRIG\_TIMD\_CMP1
- LL\_HRTIM\_BM\_TRIG\_TIMD\_CMP2
- LL\_HRTIM\_BM\_TRIG\_TIME\_RESET
- LL\_HRTIM\_BM\_TRIG\_TIME\_REPETITION
- LL\_HRTIM\_BM\_TRIG\_TIME\_CMP1
- LL\_HRTIM\_BM\_TRIG\_TIME\_CMP2
- LL\_HRTIM\_BM\_TRIG\_TIMA\_EVENT7
- LL\_HRTIM\_BM\_TRIG\_TIMD\_EVENT8
- LL\_HRTIM\_BM\_TRIG\_EVENT\_7
- LL\_HRTIM\_BM\_TRIG\_EVENT\_8
- LL\_HRTIM\_BM\_TRIG\_EVENT\_ONCHIP

Reference Manual to  
LL API cross  
reference:

- BMTRGR SW LL\_HRTIM\_BM\_GetTrig
- BMTRGR MSTRST LL\_HRTIM\_BM\_GetTrig
- BMTRGR MSTREP LL\_HRTIM\_BM\_GetTrig
- BMTRGR MSTCMP1 LL\_HRTIM\_BM\_GetTrig
- BMTRGR MSTCMP2 LL\_HRTIM\_BM\_GetTrig
- BMTRGR MSTCMP3 LL\_HRTIM\_BM\_GetTrig
- BMTRGR MSTCMP4 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TARST LL\_HRTIM\_BM\_GetTrig
- BMTRGR TAREP LL\_HRTIM\_BM\_GetTrig
- BMTRGR TACMP1 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TACMP2 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TBRST LL\_HRTIM\_BM\_GetTrig
- BMTRGR TBREP LL\_HRTIM\_BM\_GetTrig
- BMTRGR TBCMP1 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TBCMP2 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TCRST LL\_HRTIM\_BM\_GetTrig
- BMTRGR TCREP LL\_HRTIM\_BM\_GetTrig
- BMTRGR TCCMP1 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TCCMP2 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TDRST LL\_HRTIM\_BM\_GetTrig
- BMTRGR TDREP LL\_HRTIM\_BM\_GetTrig
- BMTRGR TDCMP1 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TDCMP2 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TERST LL\_HRTIM\_BM\_GetTrig
- BMTRGR TEREP LL\_HRTIM\_BM\_GetTrig
- BMTRGR TECMP1 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TECMP2 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TAEEV7 LL\_HRTIM\_BM\_GetTrig
- BMTRGR TAEEV8 LL\_HRTIM\_BM\_GetTrig
- BMTRGR EEV7 LL\_HRTIM\_BM\_GetTrig
- BMTRGR EEV8 LL\_HRTIM\_BM\_GetTrig
- BMTRGR OCHIPEV LL\_HRTIM\_BM\_GetTrig

**LL\_HRTIM\_BM\_SetCompare**

|                                                   |                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_BM_SetCompare(HRTIM_TypeDef * HRTIMx, uint32_t CompareValue)</code>                                                                                                                                                                                     |
| Function description                              | Set the burst mode controller compare value.                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>CompareValue:</b> Compare value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,...</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BMCMPR BMCMP LL_HRTIM_BM_SetCompare</li> </ul>                                                                                                                                                                                                     |

**LL\_HRTIM\_BM\_GetCompare**

|                                                   |                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_BM_GetCompare(HRTIM_TypeDef * HRTIMx)</code>                                                                                                                                               |
| Function description                              | Get actual burst mode controller compare value.                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>CompareValue:</b> Compare value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,...</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BMCMPR BMCMP LL_HRTIM_BM_GetCompare</li> </ul>                                                                                                                                            |

**LL\_HRTIM\_BM\_SetPeriod**

|                                                   |                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_BM_SetPeriod(HRTIM_TypeDef * HRTIMx, uint32_t Period)</code>                                                                                                                                                                                                                            |
| Function description                              | Set the burst mode controller period.                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Period:</b> The period value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,.... The maximum value is 0x0000 FFDF.</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BMPER BMPER LL_HRTIM_BM_SetPeriod</li> </ul>                                                                                                                                                                                                                                       |

**LL\_HRTIM\_BM\_GetPeriod**

|                      |                                                                                     |
|----------------------|-------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_BM_GetPeriod(HRTIM_TypeDef * HRTIMx)</code> |
| Function description | Get actual burst mode controller period.                                            |

|                                             |                                                                                                                                                                                                                                                          |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                  | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                          |
| Return values                               | <ul style="list-style-type: none"> <li><b>The:</b> period value must be above or equal to 3 periods of the fHRTIM clock, that is 0x60 if CKPSC[2:0] = 0, 0x30 if CKPSC[2:0] = 1, 0x18 if CKPSC[2:0] = 2,... The maximum value is 0x0000 FFDF.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>BMPER BMPER LL_HRTIM_BM_GetPeriod</li> </ul>                                                                                                                                                                      |

### LL\_HRTIM\_BM\_Enable

|                      |                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_BM_Enable(<br/>    HRTIM_TypeDef * HRTIMx)</code>           |
| Function description | Enable the burst mode controller.                                                               |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                   |

Reference Manual to LL API cross reference:

- BMCR BME LL\_HRTIM\_BM\_Enable

### LL\_HRTIM\_BM\_Disable

|                      |                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_BM_Disable(<br/>    HRTIM_TypeDef * HRTIMx)</code>          |
| Function description | Disable the burst mode controller.                                                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                   |

Reference Manual to LL API cross reference:

- BMCR BME LL\_HRTIM\_BM\_Disable

### LL\_HRTIM\_BM\_IsEnabled

|                      |                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_BM_IsEnabled(<br/>    HRTIM_TypeDef * HRTIMx)</code>                |
| Function description | Indicate whether the burst mode controller is enabled.                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> </ul>             |
| Return values        | <ul style="list-style-type: none"> <li><b>State:</b> of BME bit in HRTIM_BMCR register (1 or 0).</li> </ul> |

Reference Manual to LL API cross reference:

- BMCR BME LL\_HRTIM\_BM\_IsEnabled

**LL\_HRTIM\_BM\_Start**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_HRTIM_BM_Start (HRTIM_TypeDef * HRTIMx)</code>                       |
| Function description                              | Trigger the burst operation (software trigger)                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BMTRGR SW LL_HRTIM_BM_Start</li> </ul>                   |

**LL\_HRTIM\_BM\_Stop**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_HRTIM_BM_Stop (HRTIM_TypeDef * HRTIMx)</code>                        |
| Function description                              | Stop the burst mode operation.                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>• Causes a burst mode early termination.</li> </ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BMCR BMSTAT LL_HRTIM_BM_Stop</li> </ul>                  |

**LL\_HRTIM\_BM\_GetStatus**

|                                                   |                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_HRTIM_BM_GetStatus (HRTIM_TypeDef * HRTIMx)</code>                                                                                                                                                                        |
| Function description                              | Get actual burst mode status.                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Status:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_BM_STATUS_NORMAL</li> <li>- LL_HRTIM_BM_STATUS_BURST_ONGOING</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BMCR BMSTAT LL_HRTIM_BM_GetStatus</li> </ul>                                                                                                                                                                      |

**LL\_HRTIM\_ClearFlag\_FLT1**

|                      |                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_HRTIM_ClearFlag_FLT1 (HRTIM_TypeDef * HRTIMx)</code>                 |
| Function description | Clear the Fault 1 interrupt flag.                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |

Reference Manual to  
LL API cross  
reference:

- ICR FLT1C LL\_HRTIM\_ClearFlag\_FLT1

### **LL\_HRTIM\_IsActiveFlag\_FLT1**

Function name

**`_STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_FLT1  
(HRTIM_TypeDef * HRTIMx)`**

Function description

Indicate whether Fault 1 interrupt occurred.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **State:** of FLT1 bit in HRTIM\_ISR register (1 or 0).

Reference Manual to

LL API cross  
reference:

- ICR FLT1 LL\_HRTIM\_IsActiveFlag\_FLT1

### **LL\_HRTIM\_ClearFlag\_FLT2**

Function name

**`_STATIC_INLINE void LL_HRTIM_ClearFlag_FLT2  
(HRTIM_TypeDef * HRTIMx)`**

Function description

Clear the Fault 2 interrupt flag.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **None**

Reference Manual to

LL API cross  
reference:

- ICR FLT2C LL\_HRTIM\_ClearFlag\_FLT2

### **LL\_HRTIM\_IsActiveFlag\_FLT2**

Function name

**`_STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_FLT2  
(HRTIM_TypeDef * HRTIMx)`**

Function description

Indicate whether Fault 2 interrupt occurred.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **State:** of FLT2 bit in HRTIM\_ISR register (1 or 0).

Reference Manual to

LL API cross  
reference:

- ICR FLT2 LL\_HRTIM\_IsActiveFlag\_FLT2

### **LL\_HRTIM\_ClearFlag\_FLT3**

Function name

**`_STATIC_INLINE void LL_HRTIM_ClearFlag_FLT3  
(HRTIM_TypeDef * HRTIMx)`**

Function description

Clear the Fault 3 interrupt flag.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **None**

- ICR FLT3C LL\_HRTIM\_ClearFlag\_FLT3
- Reference Manual to  
LL API cross  
reference:

### **LL\_HRTIM\_IsActiveFlag\_FLT3**

|                                                   |                                                                                                               |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_FLT3(<br/>    HRTIM_TypeDef * HRTIMx)</code>             |
| Function description                              | Indicate whether Fault 3 interrupt occurred.                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of FLT3 bit in HRTIM_ISR register (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ICR FLT3 LL_HRTIM_IsActiveFlag_FLT3</li> </ul>                       |

### **LL\_HRTIM\_ClearFlag\_FLT4**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_FLT4(<br/>    HRTIM_TypeDef * HRTIMx)</code>        |
| Function description                              | Clear the Fault 4 interrupt flag.                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ICR FLT4C LL_HRTIM_ClearFlag_FLT4</li> </ul>             |

### **LL\_HRTIM\_IsActiveFlag\_FLT4**

|                                                   |                                                                                                               |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_FLT4(<br/>    HRTIM_TypeDef * HRTIMx)</code>             |
| Function description                              | Indicate whether Fault 4 interrupt occurred.                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of FLT4 bit in HRTIM_ISR register (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ICR FLT4 LL_HRTIM_IsActiveFlag_FLT4</li> </ul>                       |

### **LL\_HRTIM\_ClearFlag\_FLT5**

|                      |                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_FLT5(<br/>    HRTIM_TypeDef * HRTIMx)</code>        |
| Function description | Clear the Fault 5 interrupt flag.                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |

Reference Manual to  
LL API cross  
reference:

- ICR FLT5C LL\_HRTIM\_ClearFlag\_FLT5

### **LL\_HRTIM\_IsActiveFlag\_FLT5**

Function name      **`_STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_FLT5  
(HRTIM_TypeDef * HRTIMx)`**

Function description      Indicate whether Fault 5 interrupt occurred.

Parameters      • **HRTIMx:** High Resolution Timer instance

Return values      • **State:** of FLT5 bit in HRTIM\_ISR register (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ICR FLT5 LL\_HRTIM\_IsActiveFlag\_FLT5

### **LL\_HRTIM\_ClearFlag\_SYSFLT**

Function name      **`_STATIC_INLINE void LL_HRTIM_ClearFlag_SYSFLT  
(HRTIM_TypeDef * HRTIMx)`**

Function description      Clear the System Fault interrupt flag.

Parameters      • **HRTIMx:** High Resolution Timer instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:

- ISR SYSFLTC LL\_HRTIM\_ClearFlag\_SYSFLT

### **LL\_HRTIM\_IsActiveFlag\_SYSFLT**

Function name      **`_STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_SYSFLT  
(HRTIM_TypeDef * HRTIMx)`**

Function description      Indicate whether System Fault interrupt occurred.

Parameters      • **HRTIMx:** High Resolution Timer instance

Return values      • **State:** of SYSFLT bit in HRTIM\_ISR register (1 or 0).

Reference Manual to  
LL API cross  
reference:

- ISR SYSFLT LL\_HRTIM\_IsActiveFlag\_SYSFLT

### **LL\_HRTIM\_ClearFlag\_DLLRDY**

Function name      **`_STATIC_INLINE void LL_HRTIM_ClearFlag_DLLRDY  
(HRTIM_TypeDef * HRTIMx)`**

Function description      Clear the DLL ready interrupt flag.

Parameters      • **HRTIMx:** High Resolution Timer instance

Return values      • **None**

- ICR DLLRDYC LL\_HRTIM\_ClearFlag\_DLLRDY
- Reference Manual to  
LL API cross  
reference:

### **LL\_HRTIM\_IsActiveFlag\_DLLRDY**

|                                                   |                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_DLLRDY(<br/>    HRTIM_TypeDef * HRTIMx)</code>             |
| Function description                              | Indicate whether DLL ready interrupt occurred.                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of DLLRDY bit in HRTIM_ISR register (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR DLLRDY LL_HRTIM_IsActiveFlag_DLLRDY</li> </ul>                     |

### **LL\_HRTIM\_ClearFlag\_BMPER**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_BMPER(<br/>    HRTIM_TypeDef * HRTIMx)</code>       |
| Function description                              | Clear the Burst Mode period interrupt flag.                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ICR BMPERC LL_HRTIM_ClearFlag_BMPER</li> </ul>           |

### **LL\_HRTIM\_IsActiveFlag\_BMPER**

|                                                   |                                                                                                                |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_BMPER(<br/>    HRTIM_TypeDef * HRTIMx)</code>             |
| Function description                              | Indicate whether Burst Mode period interrupt occurred.                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of BMPER bit in HRTIM_ISR register (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR BMPER LL_HRTIM_IsActiveFlag_BMPER</li> </ul>                      |

### **LL\_HRTIM\_ClearFlag\_SYNC**

|                      |                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_SYNC(<br/>    HRTIM_TypeDef * HRTIMx)</code>        |
| Function description | Clear the Synchronization Input interrupt flag.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |

Reference Manual to  
LL API cross  
reference:

- MICR SYNCC LL\_HRTIM\_ClearFlag\_SYNC

### **LL\_HRTIM\_IsActiveFlag\_SYNC**

|                                                   |                                                                                                                |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_SYNC(HRTIM_TypeDef * HRTIMx)</code></b>                 |
| Function description                              | Indicate whether the Synchronization Input interrupt occurred.                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of SYNC bit in HRTIM_MISR register (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MISR SYNC LL_HRTIM_IsActiveFlag_SYNC</li> </ul>                       |

### **LL\_HRTIM\_ClearFlag\_UPDATE**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_ClearFlag_UPDATE(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                            |
| Function description                              | Clear the update interrupt flag for a given timer (including the master timer).                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MICR MUPDC LL_HRTIM_ClearFlag_UPDATE</li> <li>• TIMxICR UPDC LL_HRTIM_ClearFlag_UPDATE</li> </ul>                                                                                                                                                                                                                                                           |

### **LL\_HRTIM\_IsActiveFlag\_UPDATE**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_UPDATE(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                     |
| Function description | Indicate whether the update interrupt has occurred for a given timer (including the master timer).                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

- Return values
- **State:** of MUPD/UPD bit in HRTIM\_MISR/HRTIM\_TIMxISR register (1 or 0).

- Reference Manual to  
LL API cross  
reference:
- MISR MUPD LL\_HRTIM\_IsActiveFlag\_UPDATE
  - TIMxISR UPD LL\_HRTIM\_IsActiveFlag\_UPDATE

### **LL\_HRTIM\_ClearFlag REP**

Function name      **`_STATIC_INLINE void LL_HRTIM_ClearFlag REP  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description      Clear the repetition interrupt flag for a given timer (including the master timer).

- Parameters
- **HRTIMx:** High Resolution Timer instance
  - **Timer:** This parameter can be one of the following values:
    - LL\_HRTIM\_TIMER\_MASTER
    - LL\_HRTIM\_TIMER\_A
    - LL\_HRTIM\_TIMER\_B
    - LL\_HRTIM\_TIMER\_C
    - LL\_HRTIM\_TIMER\_D
    - LL\_HRTIM\_TIMER\_E

- Return values
- **None**

- Reference Manual to  
LL API cross  
reference:
- MICR MREPC LL\_HRTIM\_ClearFlag REP
  - TIMxICR REPC LL\_HRTIM\_ClearFlag REP

### **LL\_HRTIM\_IsActiveFlag REP**

Function name      **`_STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag REP  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description      Indicate whether the repetition interrupt has occurred for a given timer (including the master timer).

- Parameters
- **HRTIMx:** High Resolution Timer instance
  - **Timer:** This parameter can be one of the following values:
    - LL\_HRTIM\_TIMER\_MASTER
    - LL\_HRTIM\_TIMER\_A
    - LL\_HRTIM\_TIMER\_B
    - LL\_HRTIM\_TIMER\_C
    - LL\_HRTIM\_TIMER\_D
    - LL\_HRTIM\_TIMER\_E

- Return values
- **State:** of MREP/REP bit in HRTIM\_MISR/HRTIM\_TIMxISR register (1 or 0).

- Reference Manual to  
LL API cross  
reference:
- MISR MREP LL\_HRTIM\_IsActiveFlag REP
  - TIMxISR REP LL\_HRTIM\_IsActiveFlag REP

**LL\_HRTIM\_ClearFlag\_CMP1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_CMP1<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                               |
| Function description                              | Clear the compare 1 match interrupt for a given timer (including the master timer).                                                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MICR MCMP1C LL_HRTIM_ClearFlag_CMP1</li> <li>• TIMxICR CMP1C LL_HRTIM_ClearFlag_CMP1</li> </ul>                                                                                                                                                                                                                                                             |

**LL\_HRTIM\_IsActiveFlag\_CMP1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_CMP1<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                        |
| Function description                              | Indicate whether the compare match 1 interrupt has occurred for a given timer (including the master timer).                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCMP1/CMP1 bit in HRTIM_MISR/HRTIM_TIMxISR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MISR MCMP1 LL_HRTIM_IsActiveFlag_CMP1</li> <li>• TIMxISR CMP1 LL_HRTIM_IsActiveFlag_CMP1</li> </ul>                                                                                                                                                                                                                                                         |

**LL\_HRTIM\_ClearFlag\_CMP2**

|                      |                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_CMP2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                               |
| Function description | Clear the compare 2 match interrupt for a given timer (including the master timer).                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MICR MCMP2C LL_HRTIM_ClearFlag_CMP2</li> <li>• TIMxICR CMP2C LL_HRTIM_ClearFlag_CMP2</li> </ul>                 |

### LL\_HRTIM\_IsActiveFlag\_CMP2

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_CMP2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                        |
| Function description                              | Indicate whether the compare match 2 interrupt has occurred for a given timer (including the master timer).                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCMP2/CMP2 bit in HRTIM_MISR/HRTIM_TIMxISR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MISR MCMP2 LL_HRTIM_IsActiveFlag_CMP2</li> <li>• TIMxISR CMP2 LL_HRTIM_IsActiveFlag_CMP2</li> </ul>                                                                                                                                                                                                                                                         |

### LL\_HRTIM\_ClearFlag\_CMP3

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_CMP3<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                               |
| Function description                              | Clear the compare 3 match interrupt for a given timer (including the master timer).                                                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MICR MCMP3C LL_HRTIM_ClearFlag_CMP3</li> <li>• TIMxICR CMP3C LL_HRTIM_ClearFlag_CMP3</li> </ul>                                                                                                                                                                                                                                                             |

**LL\_HRTIM\_IsActiveFlag\_CMP3**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_CMP3<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                        |
| Function description                        | Indicate whether the compare match 3 interrupt has occurred for a given timer (including the master timer).                                                                                                                                                                                                                                                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCMP3/CMP3 bit in HRTIM_MISR/HRTIM_TIMxISR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                   |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MISR MCMP3 LL_HRTIM_IsActiveFlag_CMP3</li> <li>• TIMxISR CMP3 LL_HRTIM_IsActiveFlag_CMP3</li> </ul>                                                                                                                                                                                                                                                         |

**LL\_HRTIM\_ClearFlag\_CMP4**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_CMP4<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                               |
| Function description                        | Clear the compare 4 match interrupt for a given timer (including the master timer).                                                                                                                                                                                                                                                                                                                  |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MICR MCMP4C LL_HRTIM_ClearFlag_CMP4</li> <li>• TIMxICR CMP4C LL_HRTIM_ClearFlag_CMP4</li> </ul>                                                                                                                                                                                                                                                             |

**LL\_HRTIM\_IsActiveFlag\_CMP4**

|                      |                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_CMP4<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                        |
| Function description | Indicate whether the compare match 4 interrupt has occurred for a given timer (including the master timer).                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> </ul> </li> </ul> |

|                                             |                                                                                                                                                          |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                             | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCMP4/CMP4 bit in HRTIM_MISR/HRTIM_TIMxISR register (1 or 0).</li> </ul>                       |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MISR MCMP4 LL_HRTIM_IsActiveFlag_CMP4</li> <li>• TIMxISR CMP4 LL_HRTIM_IsActiveFlag_CMP4</li> </ul>             |

### LL\_HRTIM\_ClearFlag\_CPT1

|                                             |                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_CPT1(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                    |
| Function description                        | Clear the capture 1 interrupt flag for a given timer.                                                                                                                                                                                                                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                               |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• TIMxICR CPT1C LL_HRTIM_ClearFlag_CPT1</li> </ul>                                                                                                                                                                                                                                                                                     |

### LL\_HRTIM\_IsActiveFlag\_CPT1

|                                             |                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_CPT1(<br/>    HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                             |
| Function description                        | Indicate whether the capture 1 interrupt occurred for a given timer.                                                                                                                                                                                                                                                                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of CPT1 bit in HRTIM_TIMxISR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                             |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• TIMxISR CPT1 LL_HRTIM_IsActiveFlag_CPT1</li> </ul>                                                                                                                                                                                                                                                                                   |

**LL\_HRTIM\_ClearFlag\_CPT2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_CPT2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                              |
| Function description                              | Clear the capture 2 interrupt flag for a given timer.                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxICR CPT2C LL_HRTIM_ClearFlag_CPT2</li> </ul>                                                                                                                                                                                                                                                                           |

**LL\_HRTIM\_IsActiveFlag\_CPT2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_CPT2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                       |
| Function description                              | Indicate whether the capture 2 interrupt occurred for a given timer.                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of CPT2 bit in HRTIM_TIMxISR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxISR CPT2 LL_HRTIM_IsActiveFlag_CPT2</li> </ul>                                                                                                                                                                                                                                                                         |

**LL\_HRTIM\_ClearFlag\_SET1**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_SET1<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                              |
| Function description | Clear the output 1 set interrupt flag for a given timer.                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |

Reference Manual to  
LL API cross  
reference:

- TIMxICR SET1C LL\_HRTIM\_ClearFlag\_SET1

### **LL\_HRTIM\_IsActiveFlag\_SET1**

Function name

**`_STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_SET1  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Indicate whether the output 1 set interrupt occurred for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **State:** of SETx1 bit in HRTIM\_TIMxISR register (1 or 0).

Reference Manual to  
LL API cross  
reference:

- TIMxISR SET1 LL\_HRTIM\_IsActiveFlag\_SET1

### **LL\_HRTIM\_ClearFlag\_RST1**

Function name

**`_STATIC_INLINE void LL_HRTIM_ClearFlag_RST1  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Clear the output 1 reset interrupt flag for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- TIMxICR RST1C LL\_HRTIM\_ClearFlag\_RST1

### **LL\_HRTIM\_IsActiveFlag\_RST1**

Function name

**`_STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_RST1  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Indicate whether the output 1 reset interrupt occurred for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B

---

|                                                   |                                                                                                                              |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of RSTx1 bit in HRTIM_TIMxISR register (1 or 0).</li> </ul>           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxISR RST1 LL_HRTIM_IsActiveFlag_RST1</li> </ul>                                  |

### LL\_HRTIM\_ClearFlag\_SET2

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_SET2(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                   |
| Function description                              | Clear the output 2 set interrupt flag for a given timer.                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxICR SET2C LL_HRTIM_ClearFlag_SET2</li> </ul>                                                                                                                                                                                                                                                                           |

### LL\_HRTIM\_IsActiveFlag\_SET2

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_SET2(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                            |
| Function description                              | Indicate whether the output 2 set interrupt occurred for a given timer.                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of SETx2 bit in HRTIM_TIMxISR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxISR SET2 LL_HRTIM_IsActiveFlag_SET2</li> </ul>                                                                                                                                                                                                                                                                         |

### LL\_HRTIM\_ClearFlag\_RST2

|               |                                                                                                   |
|---------------|---------------------------------------------------------------------------------------------------|
| Function name | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_RST2(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code> |
|---------------|---------------------------------------------------------------------------------------------------|

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function description                              | Clear the output 2reset interrupt flag for a given timer.                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | • <b>None</b>                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | • TIMxICR RST2C LL_HRTIM_ClearFlag_RST2                                                                                                                                                                                                                                                                                                                             |

### LL\_HRTIM\_IsActiveFlag\_RST2

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_RST2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                 |
| Function description                              | Indicate whether the output 2 reset interrupt occurred for a given timer.                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | • <b>State:</b> of RSTx2 bit in HRTIM_TIMxISR register (1 or 0).                                                                                                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | • TIMxISR RST2 LL_HRTIM_IsActiveFlag_RST2                                                                                                                                                                                                                                                                                                                           |

### LL\_HRTIM\_ClearFlag\_RST

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_ClearFlag_RST<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                         |
| Function description                              | Clear the reset and/or roll-over interrupt flag for a given timer.                                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | • <b>None</b>                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | • TIMxICR RSTC LL_HRTIM_ClearFlag_RST                                                                                                                                                                                                                                                                                                                               |

**LL\_HRTIM\_IsActiveFlag\_RST**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_RST<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                        |
| Function description                              | Indicate whether the reset and/or roll-over interrupt occurred for a given timer.                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of RST bit in HRTIM_TIMxISR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxISR RST LL_HRTIM_IsActiveFlag_RST</li> </ul>                                                                                                                                                                                                                                                                           |

**LL\_HRTIM\_ClearFlag\_DLYPRT**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_ClearFlag_DLYPRT<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                            |
| Function description                              | Clear the delayed protection interrupt flag for a given timer.                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxICR DLYPRTC LL_HRTIM_ClearFlag_DLYPRT</li> </ul>                                                                                                                                                                                                                                                                       |

**LL\_HRTIM\_IsActiveFlag\_DLYPRT**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsActiveFlag_DLYPRT<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                     |
| Function description | Indicate whether the delayed protection interrupt occurred for a given timer.                                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

- |                                                              |                                                                                                                                                                            |
|--------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values<br>Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• <b>State:</b> of DLYPRT bit in HRTIM_TIMxISR register (1 or 0).</li> <li>• TIMxISR DLYPRT LL_HRTIM_IsActiveFlag_DLYPRT</li> </ul> |
|--------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### **LL\_HRTIM\_EnableIT\_FLT1**

|                                                                                                                     |                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name<br>Function description<br>Parameters<br>Return values<br>Reference Manual to LL API cross reference: | <code>_STATIC_INLINE void LL_HRTIM_EnableIT_FLT1(HRTIM_TypeDef * HRTIMx)</code><br>Enable the fault 1 interrupt.<br><ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>None</b></li> </ul> <ul style="list-style-type: none"> <li>• IER FLT1IE LL_HRTIM_EnableIT_FLT1</li> </ul> |
|---------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### **LL\_HRTIM\_DisableIT\_FLT1**

|                                                                                                                     |                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name<br>Function description<br>Parameters<br>Return values<br>Reference Manual to LL API cross reference: | <code>_STATIC_INLINE void LL_HRTIM_DisableIT_FLT1(HRTIM_TypeDef * HRTIMx)</code><br>Disable the fault 1 interrupt.<br><ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>None</b></li> </ul> <ul style="list-style-type: none"> <li>• IER FLT1IE LL_HRTIM_DisableIT_FLT1</li> </ul> |
|---------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### **LL\_HRTIM\_IsEnabledIT\_FLT1**

|                                                                                                                     |                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name<br>Function description<br>Parameters<br>Return values<br>Reference Manual to LL API cross reference: | <code>_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_FLT1(HRTIM_TypeDef * HRTIMx)</code><br>Indicate whether the fault 1 interrupt is enabled.<br><ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>State:</b> of FLT1IE bit in HRTIM_IER register (1 or 0).</li> </ul> <ul style="list-style-type: none"> <li>• IER FLT1IE LL_HRTIM_IsEnabledIT_FLT1</li> </ul> |
|---------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### **LL\_HRTIM\_EnableIT\_FLT2**

|                                                                      |                                                                                                                                                                                                                                              |
|----------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name<br>Function description<br>Parameters<br>Return values | <code>_STATIC_INLINE void LL_HRTIM_EnableIT_FLT2(HRTIM_TypeDef * HRTIMx)</code><br>Enable the fault 2 interrupt.<br><ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>None</b></li> </ul> |
|----------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Reference Manual to  
LL API cross  
reference:

- IER FLT2IE LL\_HRTIM\_EnableIT\_FLT2

### **LL\_HRTIM\_DisableIT\_FLT2**

Function name

**\_STATIC\_INLINE void LL\_HRTIM\_DisableIT\_FLT2  
(HRTIM\_TypeDef \* HRTIMx)**

Function description

Disable the fault 2 interrupt.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **None**

Reference Manual to

LL API cross  
reference:

- IER FLT2IE LL\_HRTIM\_DisableIT\_FLT2

### **LL\_HRTIM\_IsEnabledIT\_FLT2**

Function name

**\_STATIC\_INLINE uint32\_t LL\_HRTIM\_IsEnabledIT\_FLT2  
(HRTIM\_TypeDef \* HRTIMx)**

Function description

Indicate whether the fault 2 interrupt is enabled.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **State:** of FLT2IE bit in HRTIM\_IER register (1 or 0).

Reference Manual to

LL API cross  
reference:

- IER FLT2IE LL\_HRTIM\_IsEnabledIT\_FLT2

### **LL\_HRTIM\_EnableIT\_FLT3**

Function name

**\_STATIC\_INLINE void LL\_HRTIM\_EnableIT\_FLT3  
(HRTIM\_TypeDef \* HRTIMx)**

Function description

Enable the fault 3 interrupt.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **None**

Reference Manual to

LL API cross  
reference:

- IER FLT3IE LL\_HRTIM\_EnableIT\_FLT3

### **LL\_HRTIM\_DisableIT\_FLT3**

Function name

**\_STATIC\_INLINE void LL\_HRTIM\_DisableIT\_FLT3  
(HRTIM\_TypeDef \* HRTIMx)**

Function description

Disable the fault 3 interrupt.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- IER FLT3IE LL\_HRTIM\_DisableIT\_FLT3

### **LL\_HRTIM\_IsEnabledIT\_FLT3**

|                                                   |                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_FLT3(<br/>(HRTIM_TypeDef * HRTIMx)</code></b>             |
| Function description                              | Indicate whether the fault 3 interrupt is enabled.                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of FLT3IE bit in HRTIM_IER register (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER FLT3IE LL_HRTIM_IsEnabledIT_FLT3</li> </ul>                        |

### **LL\_HRTIM\_EnableIT\_FLT4**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_EnableIT_FLT4(<br/>(HRTIM_TypeDef * HRTIMx)</code></b>      |
| Function description                              | Enable the fault 4 interrupt.                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER FLT4IE LL_HRTIM_EnableIT_FLT4</li> </ul>             |

### **LL\_HRTIM\_DisableIT\_FLT4**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_DisableIT_FLT4(<br/>(HRTIM_TypeDef * HRTIMx)</code></b>     |
| Function description                              | Disable the fault 4 interrupt.                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER FLT4IE LL_HRTIM_DisableIT_FLT4</li> </ul>            |

### **LL\_HRTIM\_IsEnabledIT\_FLT4**

|                      |                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_FLT4(<br/>(HRTIM_TypeDef * HRTIMx)</code></b>             |
| Function description | Indicate whether the fault 4 interrupt is enabled.                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of FLT4IE bit in HRTIM_IER register (1 or 0).</li> </ul> |

- Reference Manual to • IER FLT4IE LL\_HRTIM\_IsEnabledIT\_FLT4  
 LL API cross reference:

### **LL\_HRTIM\_EnableIT\_FLT5**

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_HRTIM_EnableIT_FLT5(HRTIM_TypeDef * HRTIMx)</code></b> |
| Function description | Enable the fault 5 interrupt.                                                          |
| Parameters           | • <b>HRTIMx:</b> High Resolution Timer instance                                        |
| Return values        | • <b>None</b>                                                                          |

Reference Manual to • IER FLT5IE LL\_HRTIM\_EnableIT\_FLT5  
 LL API cross reference:

### **LL\_HRTIM\_DisableIT\_FLT5**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_HRTIM_DisableIT_FLT5(HRTIM_TypeDef * HRTIMx)</code></b> |
| Function description | Disable the fault 5 interrupt.                                                          |
| Parameters           | • <b>HRTIMx:</b> High Resolution Timer instance                                         |
| Return values        | • <b>None</b>                                                                           |

Reference Manual to • IER FLT5IE LL\_HRTIM\_DisableIT\_FLT5  
 LL API cross reference:

### **LL\_HRTIM\_IsEnabledIT\_FLT5**

|                      |                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_FLT5(HRTIM_TypeDef * HRTIMx)</code></b> |
| Function description | Indicate whether the fault 5 interrupt is enabled.                                            |
| Parameters           | • <b>HRTIMx:</b> High Resolution Timer instance                                               |
| Return values        | • <b>State:</b> of FLT5IE bit in HRTIM_IER register (1 or 0).                                 |

Reference Manual to • IER FLT5IE LL\_HRTIM\_IsEnabledIT\_FLT5  
 LL API cross reference:

### **LL\_HRTIM\_EnableIT\_SYSFLT**

|                      |                                                                                          |
|----------------------|------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_HRTIM_EnableIT_SYSFLT(HRTIM_TypeDef * HRTIMx)</code></b> |
| Function description | Enable the system fault interrupt.                                                       |
| Parameters           | • <b>HRTIMx:</b> High Resolution Timer instance                                          |
| Return values        | • <b>None</b>                                                                            |

- Reference Manual to LL API cross reference:
- IER SYSFLTIE LL\_HRTIM\_EnableIT\_SYSFLT

### **LL\_HRTIM\_DisableIT\_SYSFLT**

|                      |                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_HRTIM_DisableIT_SYSFLT(HRTIM_TypeDef * HRTIMx)</code></b>         |
| Function description | Disable the system fault interrupt.                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |

Reference Manual to LL API cross reference:

- IER SYSFLTIE LL\_HRTIM\_DisableIT\_SYSFLT

### **LL\_HRTIM\_IsEnabledIT\_SYSFLT**

|                      |                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_SYSFLT(HRTIM_TypeDef * HRTIMx)</code></b>                   |
| Function description | Indicate whether the system fault interrupt is enabled.                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                 |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of SYSFLTIE bit in HRTIM_IER register (1 or 0).</li> </ul> |

Reference Manual to LL API cross reference:

- IER SYSFLTIE LL\_HRTIM\_IsEnabledIT\_SYSFLT

### **LL\_HRTIM\_EnableIT\_DLLRDY**

|                      |                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_HRTIM_EnableIT_DLLRDY(HRTIM_TypeDef * HRTIMx)</code></b>          |
| Function description | Enable the DLL ready interrupt.                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |

Reference Manual to LL API cross reference:

- IER DLLRDYIE LL\_HRTIM\_EnableIT\_DLLRDY

### **LL\_HRTIM\_DisableIT\_DLLRDY**

|                      |                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_HRTIM_DisableIT_DLLRDY(HRTIM_TypeDef * HRTIMx)</code></b>         |
| Function description | Disable the DLL ready interrupt.                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- IER DLLRDYIE LL\_HRTIM\_DisableIT\_DLLRDY

### **LL\_HRTIM\_IsEnabledIT\_DLLRDY**

|                                                   |                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_DLLRDY(<br/>    HRTIM_TypeDef * HRTIMx)</code>                |
| Function description                              | Indicate whether the DLL ready interrupt is enabled.                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of DLLRDYIE bit in HRTIM_IER register (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER DLLRDYIE LL_HRTIM_IsEnabledIT_DLLRDY</li> </ul>                      |

### **LL\_HRTIM\_EnableIT\_BMPER**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_EnableIT_BMPER(<br/>    HRTIM_TypeDef * HRTIMx)</code>        |
| Function description                              | Enable the burst mode period interrupt.                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER BMPERIE LL_HRTIM_EnableIT_BMPER</li> </ul>           |

### **LL\_HRTIM\_DisableIT\_BMPER**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableIT_BMPER(<br/>    HRTIM_TypeDef * HRTIMx)</code>       |
| Function description                              | Disable the burst mode period interrupt.                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• IER BMPERIE LL_HRTIM_DisableIT_BMPER</li> </ul>          |

### **LL\_HRTIM\_IsEnabledIT\_BMPER**

|                      |                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_BMPER(<br/>    HRTIM_TypeDef * HRTIMx)</code>                |
| Function description | Indicate whether the burst mode period interrupt is enabled.                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of BMPERIE bit in HRTIM_IER register (1 or 0).</li> </ul> |

Reference Manual to  
LL API cross  
reference:

- IER BMPERIE LL\_HRTIM\_IsEnabledIT\_BMPER

### **LL\_HRTIM\_EnableIT\_SYNC**

Function name

**`_STATIC_INLINE void LL_HRTIM_EnableIT_SYNC  
(HRTIM_TypeDef * HRTIMx)`**

Function description

Enable the synchronization input interrupt.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **None**

Reference Manual to

LL API cross  
reference:

- MDIER SYNCIE LL\_HRTIM\_EnableIT\_SYNC

### **LL\_HRTIM\_DisableIT\_SYNC**

Function name

**`_STATIC_INLINE void LL_HRTIM_DisableIT_SYNC  
(HRTIM_TypeDef * HRTIMx)`**

Function description

Disable the synchronization input interrupt.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **None**

Reference Manual to

LL API cross  
reference:

- MDIER SYNCIE LL\_HRTIM\_DisableIT\_SYNC

### **LL\_HRTIM\_IsEnabledIT\_SYNC**

Function name

**`_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_SYNC  
(HRTIM_TypeDef * HRTIMx)`**

Function description

Indicate whether the synchronization input interrupt is enabled.

Parameters

- **HRTIMx:** High Resolution Timer instance

Return values

- **State:** of SYNCIE bit in HRTIM\_MDIER register (1 or 0).

Reference Manual to

LL API cross  
reference:

- MDIER SYNCIE LL\_HRTIM\_IsEnabledIT\_SYNC

### **LL\_HRTIM\_EnableIT\_UPDATE**

Function name

**`_STATIC_INLINE void LL_HRTIM_EnableIT_UPDATE  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Enable the update interrupt for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B

- LL\_HRTIM\_TIMER\_C
- LL\_HRTIM\_TIMER\_D
- LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- MDIER MUPDIE LL\_HRTIM\_EnableIT\_UPDATE
- TIMxDIER UPDIE LL\_HRTIM\_EnableIT\_UPDATE

### **LL\_HRTIM\_DisableIT\_UPDATE**

Function name

**`__STATIC_INLINE void LL_HRTIM_DisableIT_UPDATE(  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Disable the update interrupt for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- MDIER MUPDIE LL\_HRTIM\_DisableIT\_UPDATE
- TIMxDIER UPDIE LL\_HRTIM\_DisableIT\_UPDATE

### **LL\_HRTIM\_IsEnabledIT\_UPDATE**

Function name

**`__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_UPDATE(  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Indicate whether the update interrupt is enabled for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **State:** of MUPDIE/UPDIE bit in  
HRTIM\_MDIER/HRTIM\_TIMxDIER register (1 or 0).

Reference Manual to  
LL API cross  
reference:

- MDIER MUPDIE LL\_HRTIM\_IsEnabledIT\_UPDATE
- TIMxDIER UPDIE LL\_HRTIM\_IsEnabledIT\_UPDATE

**LL\_HRTIM\_EnableIT\_REP**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_EnableIT_REP<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                                 |
| Function description                              | Enable the repetition interrupt for a given timer.                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MREPIE LL_HRTIM_EnableIT_REP</li> <li>• TIMxDIER REPIE LL_HRTIM_EnableIT_REP</li> </ul>                                                                                                                                                                                                                                                               |

**LL\_HRTIM\_DisableIT\_REP**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableIT_REP<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                                |
| Function description                              | Disable the repetition interrupt for a given timer.                                                                                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MREPIE LL_HRTIM_DisableIT_REP</li> <li>• TIMxDIER REPIE LL_HRTIM_DisableIT_REP</li> </ul>                                                                                                                                                                                                                                                             |

**LL\_HRTIM\_IsEnabledIT\_REP**

|                      |                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_REP<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                              |
| Function description | Indicate whether the repetition interrupt is enabled for a given timer.                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> </ul> </li> </ul> |

- LL\_HRTIM\_TIMER\_E
- Return values**
- **State:** of MREPIE/REPIE bit in HRTIM\_MDIER/HRTIM\_TIMxDIER register (1 or 0).
- Reference Manual to LL API cross reference:**
- MDIER MREPIE LL\_HRTIM\_IsEnabledIT REP
  - TIMxDIER REPIE LL\_HRTIM\_IsEnabledIT REP

### LL\_HRTIM\_EnableIT\_CMP1

- Function name**
- ```
__STATIC_INLINE void LL_HRTIM_EnableIT_CMP1
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)
```
- Function description**
- Enable the compare 1 interrupt for a given timer.
- Parameters**
- **HRTIMx:** High Resolution Timer instance
 - **Timer:** This parameter can be one of the following values:
 - LL_HRTIM_TIMER_MASTER
 - LL_HRTIM_TIMER_A
 - LL_HRTIM_TIMER_B
 - LL_HRTIM_TIMER_C
 - LL_HRTIM_TIMER_D
 - LL_HRTIM_TIMER_E
- Return values**
- **None**
- Reference Manual to LL API cross reference:**
- MDIER MCMP1IE LL_HRTIM_EnableIT_CMP1
 - TIMxDIER CMP1IE LL_HRTIM_EnableIT_CMP1

LL_HRTIM_DisableIT_CMP1

- Function name**
- ```
__STATIC_INLINE void LL_HRTIM_DisableIT_CMP1
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)
```
- Function description**
- Disable the compare 1 interrupt for a given timer.
- Parameters**
- **HRTIMx:** High Resolution Timer instance
  - **Timer:** This parameter can be one of the following values:
    - LL\_HRTIM\_TIMER\_MASTER
    - LL\_HRTIM\_TIMER\_A
    - LL\_HRTIM\_TIMER\_B
    - LL\_HRTIM\_TIMER\_C
    - LL\_HRTIM\_TIMER\_D
    - LL\_HRTIM\_TIMER\_E
- Return values**
- **None**
- Reference Manual to LL API cross reference:**
- MDIER MCMP1IE LL\_HRTIM\_DisableIT\_CMP1
  - TIMxDIER CMP1IE LL\_HRTIM\_DisableIT\_CMP1

**LL\_HRTIM\_IsEnabledIT\_CMP1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_CMP1<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</b>                                                                                                                                                                                                                                                                                                |
| Function description                              | Indicate whether the compare 1 interrupt is enabled for a given timer.                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCMP1IE/CMP1IE bit in HRTIM_MDIER/HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP1IE LL_HRTIM_IsEnabledIT_CMP1</li> <li>• TIMxDIER CMP1IE LL_HRTIM_IsEnabledIT_CMP1</li> </ul>                                                                                                                                                                                                                                                     |

**LL\_HRTIM\_EnableIT\_CMP2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_HRTIM_EnableIT_CMP2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</b>                                                                                                                                                                                                                                                                                                       |
| Function description                              | Enable the compare 2 interrupt for a given timer.                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP2IE LL_HRTIM_EnableIT_CMP2</li> <li>• TIMxDIER CMP2IE LL_HRTIM_EnableIT_CMP2</li> </ul>                                                                                                                                                                                                                                                           |

**LL\_HRTIM\_DisableIT\_CMP2**

|                      |                                                                                                                                                                                                                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE void LL_HRTIM_DisableIT_CMP2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</b>                                                                                                                                                                                                                                              |
| Function description | Disable the compare 2 interrupt for a given timer.                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul>                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP2IE LL_HRTIM_DisableIT_CMP2</li> <li>• TIMxDIER CMP2IE LL_HRTIM_DisableIT_CMP2</li> </ul> |

### LL\_HRTIM\_IsEnabledIT\_CMP2

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_CMP2(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                        |
| Function description                              | Indicate whether the compare 2 interrupt is enabled for a given timer.                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCMP2IE/CMP2IE bit in HRTIM_MDIER/HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP2IE LL_HRTIM_IsEnabledIT_CMP2</li> <li>• TIMxDIER CMP2IE LL_HRTIM_IsEnabledIT_CMP2</li> </ul>                                                                                                                                                                                                                                                     |

### LL\_HRTIM\_EnableIT\_CMP3

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_EnableIT_CMP3(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                               |
| Function description                              | Enable the compare 3 interrupt for a given timer.                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP3IE LL_HRTIM_EnableIT_CMP3</li> <li>• TIMxDIER CMP3IE LL_HRTIM_EnableIT_CMP3</li> </ul>                                                                                                                                                                                                                                                           |

**LL\_HRTIM\_DisableIT\_CMP3**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableIT_CMP3<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                               |
| Function description                              | Disable the compare 3 interrupt for a given timer.                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP3IE LL_HRTIM_DisableIT_CMP3</li> <li>• TIMxDIER CMP3IE LL_HRTIM_DisableIT_CMP3</li> </ul>                                                                                                                                                                                                                                                         |

**LL\_HRTIM\_IsEnabledIT\_CMP3**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_CMP3<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                         |
| Function description                              | Indicate whether the compare 3 interrupt is enabled for a given timer.                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCMP3IE/CMP3IE bit in HRTIM_MDIER/HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP3IE LL_HRTIM_IsEnabledIT_CMP3</li> <li>• TIMxDIER CMP3IE LL_HRTIM_IsEnabledIT_CMP3</li> </ul>                                                                                                                                                                                                                                                     |

**LL\_HRTIM\_EnableIT\_CMP4**

|                      |                                                                                                                                                                                                                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_EnableIT_CMP4<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                        |
| Function description | Enable the compare 4 interrupt for a given timer.                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul>                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP4IE LL_HRTIM_EnableIT_CMP4</li> <li>• TIMxDIER CMP4IE LL_HRTIM_EnableIT_CMP4</li> </ul> |

### LL\_HRTIM\_DisableIT\_CMP4

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableIT_CMP4(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                                    |
| Function description                              | Disable the compare 4 interrupt for a given timer.                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP4IE LL_HRTIM_DisableIT_CMP4</li> <li>• TIMxDIER CMP4IE LL_HRTIM_DisableIT_CMP4</li> </ul>                                                                                                                                                                                                                                                         |

### LL\_HRTIM\_IsEnabledIT\_CMP4

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_CMP4(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                              |
| Function description                              | Indicate whether the compare 4 interrupt is enabled for a given timer.                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCMP4IE/CMP4IE bit in HRTIM_MIDER/HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP4IE LL_HRTIM_IsEnabledIT_CMP4</li> <li>• TIMxDIER CMP4IE LL_HRTIM_IsEnabledIT_CMP4</li> </ul>                                                                                                                                                                                                                                                     |

**LL\_HRTIM\_EnableIT\_CPT1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_EnableIT_CPT1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                    |
| Function description                              | Enable the capture 1 interrupt for a given timer.                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER CPT1IE LL_HRTIM_EnableIT_CPT1</li> </ul>                                                                                                                                                                                                                                                                          |

**LL\_HRTIM\_DisableIT\_CPT1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableIT_CPT1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                   |
| Function description                              | Enable the capture 1 interrupt for a given timer.                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER CPT1IE LL_HRTIM_DisableIT_CPT1</li> </ul>                                                                                                                                                                                                                                                                         |

**LL\_HRTIM\_IsEnabledIT\_CPT1**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_CPT1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                             |
| Function description | Indicate whether the capture 1 interrupt is enabled for a given timer.                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of CPT1IE bit in HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                |

- Reference Manual to  
LL API cross  
reference:
- TIMxDIER CPT1IE LL\_HRTIM\_IsEnabledIT\_CPT1

### **LL\_HRTIM\_EnableIT\_CPT2**

- Function name      **`_STATIC_INLINE void LL_HRTIM_EnableIT_CPT2  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**
- Function description      Enable the capture 2 interrupt for a given timer.
- Parameters
  - **HRTIMx:** High Resolution Timer instance
  - **Timer:** This parameter can be one of the following values:
    - LL\_HRTIM\_TIMER\_A
    - LL\_HRTIM\_TIMER\_B
    - LL\_HRTIM\_TIMER\_C
    - LL\_HRTIM\_TIMER\_D
    - LL\_HRTIM\_TIMER\_E
- Return values
  - **None**
- Reference Manual to  
LL API cross  
reference:
- TIMxDIER CPT2IE LL\_HRTIM\_EnableIT\_CPT2

### **LL\_HRTIM\_DisableIT\_CPT2**

- Function name      **`_STATIC_INLINE void LL_HRTIM_DisableIT_CPT2  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**
- Function description      Enable the capture 2 interrupt for a given timer.
- Parameters
  - **HRTIMx:** High Resolution Timer instance
  - **Timer:** This parameter can be one of the following values:
    - LL\_HRTIM\_TIMER\_A
    - LL\_HRTIM\_TIMER\_B
    - LL\_HRTIM\_TIMER\_C
    - LL\_HRTIM\_TIMER\_D
    - LL\_HRTIM\_TIMER\_E
- Return values
  - **None**
- Reference Manual to  
LL API cross  
reference:
- TIMxDIER CPT2IE LL\_HRTIM\_DisableIT\_CPT2

### **LL\_HRTIM\_IsEnabledIT\_CPT2**

- Function name      **`_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_CPT2  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**
- Function description      Indicate whether the capture 2 interrupt is enabled for a given timer.
- Parameters
  - **HRTIMx:** High Resolution Timer instance
  - **Timer:** This parameter can be one of the following values:
    - LL\_HRTIM\_TIMER\_A
    - LL\_HRTIM\_TIMER\_B

- LL\_HRTIM\_TIMER\_C
- LL\_HRTIM\_TIMER\_D
- LL\_HRTIM\_TIMER\_E

Return values

- **State:** of CPT2IE bit in HRTIM\_TIMxDIER register (1 or 0).

Reference Manual to  
LL API cross  
reference:

- TIMxDIER CPT2IE LL\_HRTIM\_IsEnabledIT\_CPT2

### **LL\_HRTIM\_EnableIT\_SET1**

Function name

```
__STATIC_INLINE void LL_HRTIM_EnableIT_SET1
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)
```

Function description

Enable the output 1 set interrupt for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- TIMxDIER SET1IE LL\_HRTIM\_EnableIT\_SET1

### **LL\_HRTIM\_DisableIT\_SET1**

Function name

```
__STATIC_INLINE void LL_HRTIM_DisableIT_SET1
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)
```

Function description

Disable the output 1 set interrupt for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- TIMxDIER SET1IE LL\_HRTIM\_DisableIT\_SET1

**LL\_HRTIM\_IsEnabledIT\_SET1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_SET1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                       |
| Function description                              | Indicate whether the output 1 set interrupt is enabled for a given timer.                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of SET1xIE bit in HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER SET1IE LL_HRTIM_IsEnabledIT_SET1</li> </ul>                                                                                                                                                                                                                                                                       |

**LL\_HRTIM\_EnableIT\_RST1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_EnableIT_RST1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                              |
| Function description                              | Enable the output 1 reset interrupt for a given timer.                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER RST1IE LL_HRTIM_EnableIT_RST1</li> </ul>                                                                                                                                                                                                                                                                          |

**LL\_HRTIM\_DisableIT\_RST1**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_HRTIM_DisableIT_RST1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                             |
| Function description | Disable the output 1 reset interrupt for a given timer.                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |

Reference Manual to  
LL API cross  
reference:

- TIMxDIER RST1IE LL\_HRTIM\_DisableIT\_RST1

### **LL\_HRTIM\_IsEnabledIT\_RST1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_RST1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                       |
| Function description                              | Indicate whether the output 1 reset interrupt is enabled for a given timer.                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of RST1xIE bit in HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER RST1IE LL_HRTIM_IsEnabledIT_RST1</li> </ul>                                                                                                                                                                                                                                                                       |

### **LL\_HRTIM\_EnableIT\_SET2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_EnableIT_SET2(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                              |
| Function description                              | Enable the output 2 set interrupt for a given timer.                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER SET2IE LL_HRTIM_EnableIT_SET2</li> </ul>                                                                                                                                                                                                                                                                          |

### **LL\_HRTIM\_DisableIT\_SET2**

|                      |                                                                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_HRTIM_DisableIT_SET2(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                         |
| Function description | Disable the output 2 set interrupt for a given timer.                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> </ul> </li> </ul> |

- LL\_HRTIM\_TIMER\_C
- LL\_HRTIM\_TIMER\_D
- LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- TIMxDIER SET2IE LL\_HRTIM\_DisableIT\_SET2

### **LL\_HRTIM\_IsEnabledIT\_SET2**

Function name

**`__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_SET2(  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Indicate whether the output 2 set interrupt is enabled for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **State:** of SET2xIE bit in HRTIM\_TIMxDIER register (1 or 0).

Reference Manual to  
LL API cross  
reference:

- TIMxDIER SET2IE LL\_HRTIM\_IsEnabledIT\_SET2

### **LL\_HRTIM\_EnableIT\_RST2**

Function name

**`__STATIC_INLINE void LL_HRTIM_EnableIT_RST2(  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Enable the output 2 reset interrupt for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- TIMxDIER RST2IE LL\_HRTIM\_EnableIT\_RST2

**LL\_HRTIM\_DisableIT\_RST2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableIT_RST2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                              |
| Function description                              | Disable the output 2 reset interrupt for a given timer.                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMx DIER RST2IE LL_HRTIM_DisableIT_RST2</li> </ul>                                                                                                                                                                                                                                                                        |

**LL\_HRTIM\_IsEnabledIT\_RST2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_RST2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                        |
| Function description                              | Indicate whether the output 2 reset LL_HRTIM_IsEnabledIT_RST2 is enabled for a given timer.                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of RST2xIE bit in HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMx DIER RST2IE LL_HRTIM_DisableIT_RST2</li> </ul>                                                                                                                                                                                                                                                                        |

**LL\_HRTIM\_EnableIT\_RST**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_EnableIT_RST<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                |
| Function description | Enable the reset/roll-over interrupt for a given timer.                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |

Reference Manual to  
LL API cross  
reference:

- TIMxDIER RSTIE LL\_HRTIM\_EnableIT\_RST

### **LL\_HRTIM\_DisableIT\_RST**

Function name

**`_STATIC_INLINE void LL_HRTIM_DisableIT_RST  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Disable the reset/roll-over interrupt for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to

LL API cross

reference:

- TIMxDIER RSTIE LL\_HRTIM\_DisableIT\_RST

### **LL\_HRTIM\_IsEnabledIT\_RST**

Function name

**`_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_RST  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Indicate whether the reset/roll-over interrupt is enabled for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **State:** of RSTIE bit in HRTIM\_TIMxDIER register (1 or 0).

Reference Manual to

LL API cross

reference:

- TIMxDIER RSTIE LL\_HRTIM\_IsEnabledIT\_RST

### **LL\_HRTIM\_EnableIT\_DLYPRT**

Function name

**`_STATIC_INLINE void LL_HRTIM_EnableIT_DLYPRT  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Enable the delayed protection interrupt for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B

- LL\_HRTIM\_TIMER\_C
- LL\_HRTIM\_TIMER\_D
- LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- TIMxDIER DLYPRTIE LL\_HRTIM\_EnableIT\_DLYPRT

**LL\_HRTIM\_DisableIT\_DLYPRT**

Function name

**`_STATIC_INLINE void LL_HRTIM_DisableIT_DLYPRT(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Disable the delayed protection interrupt for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- TIMxDIER DLYPRTIE LL\_HRTIM\_DisableIT\_DLYPRT

**LL\_HRTIM\_IsEnabledIT\_DLYPRT**

Function name

**`_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledIT_DLYPRT(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Indicate whether the delayed protection interrupt is enabled for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **State:** of DLYPRTIE bit in HRTIM\_TIMxDIER register (1 or 0).

Reference Manual to  
LL API cross  
reference:

- TIMxDIER DLYPRTIE LL\_HRTIM\_IsEnabledIT\_DLYPRT

**LL\_HRTIM\_EnableDMAReq\_SYNC**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_SYNC<br/>(HRTIM_TypeDef * HRTIMx)</code>         |
| Function description                              | Enable the synchronization input DMA request.                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER SYNCDE LL_HRTIM_EnableDMAReq_SYNC</li> </ul>       |

**LL\_HRTIM\_DisableDMAReq\_SYNC**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_SYNC<br/>(HRTIM_TypeDef * HRTIMx)</code>        |
| Function description                              | Disable the synchronization input DMA request.                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER SYNCDE LL_HRTIM_DisableDMAReq_SYNC</li> </ul>      |

**LL\_HRTIM\_IsEnabledDMAReq\_SYNC**

|                                                   |                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_IsEnabledDMAReq_SYNC (HRTIM_TypeDef *<br/>HRTIMx)</code>              |
| Function description                              | Indicate whether the synchronization input DMA request is enabled.                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of SYNCDE bit in HRTIM_MDIER register (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER SYNCDE LL_HRTIM_IsEnabledDMAReq_SYNC</li> </ul>                    |

**LL\_HRTIM\_EnableDMAReq\_UPDATE**

|                      |                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_UPDATE<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                              |
| Function description | Enable the update DMA request for a given timer.                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> </ul> </li> </ul> |

- LL\_HRTIM\_TIMER\_E

## Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- MDIER MUPDDE LL\_HRTIM\_EnableDMAReq\_UPDATE
- TIMxDIER UPDDE LL\_HRTIM\_EnableDMAReq\_UPDATE

**LL\_HRTIM\_DisableDMAReq\_UPDATE**

## Function name

**`_STATIC_INLINE void LL_HRTIM_DisableDMAReq_UPDATE (HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

## Function description

Disable the update DMA request for a given timer.

## Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

## Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- MDIER MUPDDE LL\_HRTIM\_DisableDMAReq\_UPDATE
- TIMxDIER UPDDE LL\_HRTIM\_DisableDMAReq\_UPDATE

**LL\_HRTIM\_IsEnabledDMAReq\_UPDATE**

## Function name

**`_STATIC_INLINE uint32_t LL_HRTIM_IsEnabledDMAReq_UPDATE (HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

## Function description

Indicate whether the update DMA request is enabled for a given timer.

## Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

## Return values

- **State:** of MUPDDE/UPDDE bit in HRTIM\_MDIER/HRTIM\_TIMxDIER register (1 or 0).

Reference Manual to  
LL API cross  
reference:

- MDIER MUPDDE LL\_HRTIM\_IsEnabledDMAReq\_UPDATE
- TIMxDIER UPDDE LL\_HRTIM\_IsEnabledDMAReq\_UPDATE

**LL\_HRTIM\_EnableDMAReq REP**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_EnableDMAReq REP<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                       |
| Function description                              | Enable the repetition DMA request for a given timer.                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MREPDE LL_HRTIM_EnableDMAReq REP</li> <li>• TIMxDIER REPDE LL_HRTIM_EnableDMAReq REP</li> </ul>                                                                                                                                                                                                                                                       |

**LL\_HRTIM\_DisableDMAReq REP**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_HRTIM_DisableDMAReq REP<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                      |
| Function description                              | Disable the repetition DMA request for a given timer.                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MREPDE LL_HRTIM_DisableDMAReq REP</li> <li>• TIMxDIER REPDE LL_HRTIM_DisableDMAReq REP</li> </ul>                                                                                                                                                                                                                                                     |

**LL\_HRTIM\_IsEnabledDMAReq REP**

|                      |                                                                                                                                                                                                                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t<br/>LL_HRTIM_IsEnabledDMAReq REP (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                    |
| Function description | Indicate whether the repetition DMA request is enabled for a given timer.                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> </ul> </li> </ul> |

|                                             |                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                             | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul>                                                     |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of MREPDE/REPDE bit in HRTIM_MDIER/HRTIM_TIMxDIER register (1 or 0).</li> </ul>               |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MDIER MREPDE LL_HRTIM_IsEnabledDMAReq REP</li> <li>• TIMxDIER REPDE LL_HRTIM_IsEnabledDMAReq REP</li> </ul> |

### LL\_HRTIM\_EnableDMAReq\_CMP1

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_HRTIM_EnableDMAReq_CMP1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                           |
| Function description                        | Enable the compare 1 DMA request for a given timer.                                                                                                                                                                                                                                                                                                                                                  |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP1DE LL_HRTIM_EnableDMAReq_CMP1</li> <li>• TIMxDIER CMP1DE LL_HRTIM_EnableDMAReq_CMP1</li> </ul>                                                                                                                                                                                                                                                   |

### LL\_HRTIM\_DisableDMAReq\_CMP1

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_HRTIM_DisableDMAReq_CMP1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                                                          |
| Function description                        | Disable the compare 1 DMA request for a given timer.                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_MASTER</li> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP1DE LL_HRTIM_DisableDMAReq_CMP1</li> <li>• TIMxDIER CMP1DE LL_HRTIM_DisableDMAReq_CMP1</li> </ul>                                                                                                                                                                                                                                                 |

**LL\_HRTIM\_IsEnabledDMAReq\_CMP1**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledDMAReq_CMP1 (HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                         |
| Function description                        | Indicate whether the compare 1 DMA request is enabled for a given timer.                                                                                                                                                                                                                                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCMP1DE/CMP1DE bit in HRTIM_MDIER/HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                             |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP1DE LL_HRTIM_IsEnabledDMAReq_CMP1</li> <li>• TIMxDIER CMP1DE LL_HRTIM_IsEnabledDMAReq_CMP1</li> </ul>                                                                                                                                                                                                                                             |

**LL\_HRTIM\_EnableDMAReq\_CMP2**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_CMP2 (HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                                |
| Function description                        | Enable the compare 2 DMA request for a given timer.                                                                                                                                                                                                                                                                                                                                                  |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP2DE LL_HRTIM_EnableDMAReq_CMP2</li> <li>• TIMxDIER CMP2DE LL_HRTIM_EnableDMAReq_CMP2</li> </ul>                                                                                                                                                                                                                                                   |

**LL\_HRTIM\_DisableDMAReq\_CMP2**

|                      |                                                                                                                                                                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_CMP2 (HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                           |
| Function description | Disable the compare 2 DMA request for a given timer.                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> </ul> </li> </ul> |

- LL\_HRTIM\_TIMER\_C
- LL\_HRTIM\_TIMER\_D
- LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- MDIER MCMP2DE LL\_HRTIM\_DisableDMAReq\_CMP2
- TIMxDIER CMP2DE LL\_HRTIM\_DisableDMAReq\_CMP2

### **LL\_HRTIM\_IsEnabledDMAReq\_CMP2**

Function name

```
__STATIC_INLINE uint32_t
LL_HRTIM_IsEnabledDMAReq_CMP2 (HRTIM_TypeDef *
HRTIMx, uint32_t Timer)
```

Function description

Indicate whether the compare 2 DMA request is enabled for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **State:** of MCMP2DE/CMP2DE bit in HRTIM\_MDIER/HRTIM\_TIMxDIER register (1 or 0).

Reference Manual to  
LL API cross  
reference:

- MDIER MCMP2DE LL\_HRTIM\_IsEnabledDMAReq\_CMP2
- TIMxDIER CMP2DE LL\_HRTIM\_IsEnabledDMAReq\_CMP2

### **LL\_HRTIM\_EnableDMAReq\_CMP3**

Function name

```
__STATIC_INLINE void LL_HRTIM_EnableDMAReq_CMP3
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)
```

Function description

Enable the compare 3 DMA request for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- MDIER MCMP3DE LL\_HRTIM\_EnableDMAReq\_CMP3
- TIMxDIER CMP3DE LL\_HRTIM\_EnableDMAReq\_CMP3

**LL\_HRTIM\_DisableDMAReq\_CMP3**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_CMP3<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                           |
| Function description                              | Disable the compare 3 DMA request for a given timer.                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP3DE LL_HRTIM_DisableDMAReq_CMP3</li> <li>• TIMxDIER CMP3DE LL_HRTIM_DisableDMAReq_CMP3</li> </ul>                                                                                                                                                                                                                                                 |

**LL\_HRTIM\_IsEnabledDMAReq\_CMP3**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_IsEnabledDMAReq_CMP3 (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                                 |
| Function description                              | Indicate whether the compare 3 DMA request is enabled for a given timer.                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of MCMP3DE/CMP3DE bit in HRTIM_MDIER/HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• MDIER MCMP3DE LL_HRTIM_IsEnabledDMAReq_CMP3</li> <li>• TIMxDIER CMP3DE LL_HRTIM_IsEnabledDMAReq_CMP3</li> </ul>                                                                                                                                                                                                                                             |

**LL\_HRTIM\_EnableDMAReq\_CMP4**

|                      |                                                                                                                                                                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_CMP4<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                        |
| Function description | Enable the compare 4 DMA request for a given timer.                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_MASTER</li> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> </ul> </li> </ul> |

- LL\_HRTIM\_TIMER\_C
- LL\_HRTIM\_TIMER\_D
- LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- MDIER MCMP4DE LL\_HRTIM\_EnableDMAReq\_CMP4
- TIMxDIER CMP4DE LL\_HRTIM\_EnableDMAReq\_CMP4

### **LL\_HRTIM\_DisableDMAReq\_CMP4**

Function name

**`_STATIC_INLINE void LL_HRTIM_DisableDMAReq_CMP4  
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)`**

Function description

Disable the compare 4 DMA request for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- MDIER MCMP4DE LL\_HRTIM\_DisableDMAReq\_CMP4
- TIMxDIER CMP4DE LL\_HRTIM\_DisableDMAReq\_CMP4

### **LL\_HRTIM\_IsEnabledDMAReq\_CMP4**

Function name

**`_STATIC_INLINE uint32_t  
LL_HRTIM_IsEnabledDMAReq_CMP4 (HRTIM_TypeDef *  
HRTIMx, uint32_t Timer)`**

Function description

Indicate whether the compare 4 DMA request is enabled for a given timer.

Parameters

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_MASTER
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

Return values

- **State:** of MCMP4DE/CMP4DE bit in HRTIM\_MDIER/HRTIM\_TIMxDIER register (1 or 0).

Reference Manual to  
LL API cross  
reference:

- MDIER MCMP4DE LL\_HRTIM\_IsEnabledDMAReq\_CMP4
- TIMxDIER CMP4DE LL\_HRTIM\_IsEnabledDMAReq\_CMP4

**LL\_HRTIM\_EnableDMAReq\_CPT1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_CPT1<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                           |
| Function description                              | Enable the capture 1 DMA request for a given timer.                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER CPT1DE LL_HRTIM_EnableDMAReq_CPT1</li> </ul>                                                                                                                                                                                                                                                                      |

**LL\_HRTIM\_DisableDMAReq\_CPT1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_CPT1<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                          |
| Function description                              | Disable the capture 1 DMA request for a given timer.                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER CPT1DE LL_HRTIM_DisableDMAReq_CPT1</li> </ul>                                                                                                                                                                                                                                                                     |

**LL\_HRTIM\_IsEnabledDMAReq\_CPT1**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_IsEnabledDMAReq_CPT1 (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                |
| Function description | Indicate whether the capture 1 DMA request is enabled for a given timer.                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

|                                                   |                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of CPT1DE bit in HRTIM_TIMxDIER register (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TIMxDIER CPT1DE LL_HRTIM_IsEnabledDMAReq_CPT1</li> </ul>                    |

### LL\_HRTIM\_EnableDMAReq\_CPT2

|                                                   |                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_CPT2(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                            |
| Function description                              | Enable the capture 2 DMA request for a given timer.                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TIMxDIER CPT2DE LL_HRTIM_EnableDMAReq_CPT2</li> </ul>                                                                                                                                                                                                                                                                    |

### LL\_HRTIM\_DisableDMAReq\_CPT2

|                                                   |                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_CPT2(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                           |
| Function description                              | Disable the capture 2 DMA request for a given timer.                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TIMxDIER CPT2DE LL_HRTIM_DisableDMAReq_CPT2</li> </ul>                                                                                                                                                                                                                                                                   |

### LL\_HRTIM\_IsEnabledDMAReq\_CPT2

|                      |                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledDMAReq_CPT2(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                     |
| Function description | Indicate whether the capture 2 DMA request is enabled for a given timer.                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of CPT2DE bit in HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER CPT2DE LL_HRTIM_IsEnabledDMAReq_CPT2</li> </ul>                                                                                    |

### LL\_HRTIM\_EnableDMAReq\_SET1

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_SET1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                                |
| Function description                              | Enable the output 1 set DMA request for a given timer.                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER SET1DE LL_HRTIM_EnableDMAReq_SET1</li> </ul>                                                                                                                                                                                                                                                                      |

### LL\_HRTIM\_DisableDMAReq\_SET1

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_SET1(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                               |
| Function description                              | Disable the output 1 set DMA request for a given timer.                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER SET1DE LL_HRTIM_DisableDMAReq_SET1</li> </ul>                                                                                                                                                                                                                                                                     |

**LL\_HRTIM\_IsEnabledDMAReq\_SET1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_IsEnabledDMAReq_SET1 (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                |
| Function description                              | Indicate whether the output 1 set DMA request is enabled for a given timer.                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of SET1xDE bit in HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER SET1DE LL_HRTIM_IsEnabledDMAReq_SET1</li> </ul>                                                                                                                                                                                                                                                                   |

**LL\_HRTIM\_EnableDMAReq\_RST1**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_RST1<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                           |
| Function description                              | Enable the output 1 reset DMA request for a given timer.                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER RST1DE LL_HRTIM_EnableDMAReq_RST1</li> </ul>                                                                                                                                                                                                                                                                      |

**LL\_HRTIM\_DisableDMAReq\_RST1**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_RST1<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                          |
| Function description | Disable the output 1 reset DMA request for a given timer.                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

|                                                   |                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER RST1DE LL_HRTIM_DisableDMAReq_RST1</li> </ul> |

### LL\_HRTIM\_IsEnabledDMAReq\_RST1

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_IsEnabledDMAReq_RST1 (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                |
| Function description                              | Indicate whether the output 1 reset interrupt is enabled for a given timer.                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of RST1xDE bit in HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER RST1DE LL_HRTIM_IsEnabledDMAReq_RST1</li> </ul>                                                                                                                                                                                                                                                                   |

### LL\_HRTIM\_EnableDMAReq\_SET2

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_SET2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                           |
| Function description                              | Enable the output 2 set DMA request for a given timer.                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER SET2DE LL_HRTIM_EnableDMAReq_SET2</li> </ul>                                                                                                                                                                                                                                                                      |

### LL\_HRTIM\_DisableDMAReq\_SET2

|                      |                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_SET2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                    |
| Function description | Disable the output 2 set DMA request for a given timer.                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values:</li> </ul> |

- LL\_HRTIM\_TIMER\_A
- LL\_HRTIM\_TIMER\_B
- LL\_HRTIM\_TIMER\_C
- LL\_HRTIM\_TIMER\_D
- LL\_HRTIM\_TIMER\_E

**Return values**

**Reference Manual to  
LL API cross  
reference:**

- **None**

- TIMxDIER SET2DE LL\_HRTIM\_DisableDMAReq\_SET2

### **LL\_HRTIM\_IsEnabledDMAReq\_SET2**

**Function name**

```
__STATIC_INLINE uint32_t
LL_HRTIM_IsEnabledDMAReq_SET2 (HRTIM_TypeDef *
HRTIMx, uint32_t Timer)
```

**Function description**

Indicate whether the output 2 set DMA request is enabled for a given timer.

**Parameters**

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

**Return values**

- **State:** of SET2xDE bit in HRTIM\_TIMxDIER register (1 or 0).

**Reference Manual to  
LL API cross  
reference:**

- TIMxDIER SET2DE LL\_HRTIM\_IsEnabledDMAReq\_SET2

### **LL\_HRTIM\_EnableDMAReq\_RST2**

**Function name**

```
__STATIC_INLINE void LL_HRTIM_EnableDMAReq_RST2
(HRTIM_TypeDef * HRTIMx, uint32_t Timer)
```

**Function description**

Enable the output 2 reset DMA request for a given timer.

**Parameters**

- **HRTIMx:** High Resolution Timer instance
- **Timer:** This parameter can be one of the following values:
  - LL\_HRTIM\_TIMER\_A
  - LL\_HRTIM\_TIMER\_B
  - LL\_HRTIM\_TIMER\_C
  - LL\_HRTIM\_TIMER\_D
  - LL\_HRTIM\_TIMER\_E

**Return values**

- **None**

**Reference Manual to  
LL API cross  
reference:**

- TIMxDIER RST2DE LL\_HRTIM\_EnableDMAReq\_RST2

**LL\_HRTIM\_DisableDMAReq\_RST2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_RST2<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                          |
| Function description                              | Disable the output 2 reset DMA request for a given timer.                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER RST2DE LL_HRTIM_DisableDMAReq_RST2</li> </ul>                                                                                                                                                                                                                                                                     |

**LL\_HRTIM\_IsEnabledDMAReq\_RST2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_HRTIM_IsEnabledDMAReq_RST2 (HRTIM_TypeDef *<br/>HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                |
| Function description                              | Indicate whether the output 2 reset DMA request is enabled for a given timer.                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of RST2xDE bit in HRTIM_TIMxR register (1 or 0).</li> </ul>                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER RST2DE LL_HRTIM_IsEnabledDMAReq_RST2</li> </ul>                                                                                                                                                                                                                                                                   |

**LL\_HRTIM\_EnableDMAReq\_RST**

|                      |                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_RST<br/>(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                            |
| Function description | Enable the reset/roll-over DMA request for a given timer.                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TIMxDIER RSTDE LL_HRTIM_EnableDMAReq_RST</li> </ul> |

### LL\_HRTIM\_DisableDMAReq\_RST

|                                                   |                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_RST(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                                     |
| Function description                              | Disable the reset/roll-over DMA request for a given timer.                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TIMxDIER RSTDE LL_HRTIM_DisableDMAReq_RST</li> </ul>                                                                                                                                                                                                                                                                     |

### LL\_HRTIM\_IsEnabledDMAReq\_RST

|                                                   |                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledDMAReq_RST(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                                                                                                                                                                                                               |
| Function description                              | Indicate whether the reset/roll-over DMA request is enabled for a given timer.                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_HRTIM_TIMER_A</li> <li>– LL_HRTIM_TIMER_B</li> <li>– LL_HRTIM_TIMER_C</li> <li>– LL_HRTIM_TIMER_D</li> <li>– LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of RSTDE bit in HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TIMxDIER RSTDE LL_HRTIM_IsEnabledDMAReq_RST</li> </ul>                                                                                                                                                                                                                                                                   |

### LL\_HRTIM\_EnableDMAReq\_DLYPRT

|                      |                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_HRTIM_EnableDMAReq_DLYPRT(HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code></b>                                                             |
| Function description | Enable the delayed protection DMA request for a given timer.                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>HRTIMx:</b> High Resolution Timer instance</li> <li><b>Timer:</b> This parameter can be one of the following values:</li> </ul> |

|                                                   |                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER DLYPRTDE</li> <li>  LL_HRTIM_EnableDMAReq_DLYPRT</li> </ul>                                                                        |

### LL\_HRTIM\_DisableDMAReq\_DLYPRT

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_HRTIM_DisableDMAReq_DLYPRT (HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                            |
| Function description                              | Disable the delayed protection DMA request for a given timer.                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER DLYPRTDE</li> <li>  LL_HRTIM_DisableDMAReq_DLYPRT</li> </ul>                                                                                                                                                                                                                                                      |

### LL\_HRTIM\_IsEnabledDMAReq\_DLYPRT

|                                                   |                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_HRTIM_IsEnabledDMAReq_DLYPRT (HRTIM_TypeDef * HRTIMx, uint32_t Timer)</code>                                                                                                                                                                                                                                                      |
| Function description                              | Indicate whether the delayed protection DMA request is enabled for a given timer.                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> <li>• <b>Timer:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_HRTIM_TIMER_A</li> <li>- LL_HRTIM_TIMER_B</li> <li>- LL_HRTIM_TIMER_C</li> <li>- LL_HRTIM_TIMER_D</li> <li>- LL_HRTIM_TIMER_E</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of DLYPRTDE bit in HRTIM_TIMxDIER register (1 or 0).</li> </ul>                                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMxDIER DLYPRTDE</li> <li>  LL_HRTIM_IsEnabledDMAReq_DLYPRT</li> </ul>                                                                                                                                                                                                                                                    |

**LL\_HRTIM\_DeInit**

|                      |                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_HRTIM_DeInit (HRTIM_TypeDef * HRTIMx)</b>                                                                                                                                                                                      |
| Function description | Set HRTIM instance registers to their reset values.                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HRTIMx:</b> High Resolution Timer instance</li> </ul>                                                                                                                                                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>ErrorStatus:</b> enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: HRTIMx registers are de-initialized</li> <li>– ERROR: invalid HRTIMx instance</li> </ul> </li> </ul> |

**68.2 HRTIM Firmware driver defines****68.2.1 HRTIM*****ADC TRIGGER***

|                    |                          |
|--------------------|--------------------------|
| LL_HRTIM_ADCTRIG_1 | ADC trigger 1 identifier |
| LL_HRTIM_ADCTRIG_2 | ADC trigger 2 identifier |
| LL_HRTIM_ADCTRIG_3 | ADC trigger 3 identifier |
| LL_HRTIM_ADCTRIG_4 | ADC trigger 4 identifier |

***ADC TRIGGER 1/3 SOURCE***

|                                 |                                  |
|---------------------------------|----------------------------------|
| LL_HRTIM_ADCTRIG_SRC13_NONE     | No ADC trigger event             |
| LL_HRTIM_ADCTRIG_SRC13_MCMP1    | ADC Trigger on master compare 1  |
| LL_HRTIM_ADCTRIG_SRC13_MCMP2    | ADC Trigger on master compare 2  |
| LL_HRTIM_ADCTRIG_SRC13_MCMP3    | ADC Trigger on master compare 3  |
| LL_HRTIM_ADCTRIG_SRC13_MCMP4    | ADC Trigger on master compare 4  |
| LL_HRTIM_ADCTRIG_SRC13_MPER     | ADC Trigger on master period     |
| LL_HRTIM_ADCTRIG_SRC13_EEV1     | ADC Trigger on external event 1  |
| LL_HRTIM_ADCTRIG_SRC13_EEV2     | ADC Trigger on external event 2  |
| LL_HRTIM_ADCTRIG_SRC13_EEV3     | ADC Trigger on external event 3  |
| LL_HRTIM_ADCTRIG_SRC13_EEV4     | ADC Trigger on external event 4  |
| LL_HRTIM_ADCTRIG_SRC13_EEV5     | ADC Trigger on external event 5  |
| LL_HRTIM_ADCTRIG_SRC13_TIMACMP2 | ADC Trigger on Timer A compare 2 |
| LL_HRTIM_ADCTRIG_SRC13_TIMACMP3 | ADC Trigger on Timer A compare 3 |
| LL_HRTIM_ADCTRIG_SRC13_TIMACMP4 | ADC Trigger on Timer A compare 4 |
| LL_HRTIM_ADCTRIG_SRC13_TIMAPER  | ADC Trigger on Timer A period    |
| LL_HRTIM_ADCTRIG_SRC13_TIMARST  | ADC Trigger on Timer A reset     |
| LL_HRTIM_ADCTRIG_SRC13_TIMBCMP2 | ADC Trigger on Timer B compare 2 |
| LL_HRTIM_ADCTRIG_SRC13_TIMBCMP3 | ADC Trigger on Timer B compare 3 |
| LL_HRTIM_ADCTRIG_SRC13_TIMBCMP4 | ADC Trigger on Timer B compare 4 |
| LL_HRTIM_ADCTRIG_SRC13_TIMBPER  | ADC Trigger on Timer B period    |

|                                 |                                  |
|---------------------------------|----------------------------------|
| LL_HRTIM_ADCTRIG_SRC13_TIMBRST  | ADC Trigger on Timer B reset     |
| LL_HRTIM_ADCTRIG_SRC13_TIMCCMP2 | ADC Trigger on Timer C compare 2 |
| LL_HRTIM_ADCTRIG_SRC13_TIMCCMP3 | ADC Trigger on Timer C compare 3 |
| LL_HRTIM_ADCTRIG_SRC13_TIMCCMP4 | ADC Trigger on Timer C compare 4 |
| LL_HRTIM_ADCTRIG_SRC13_TIMCPER  | ADC Trigger on Timer C period    |
| LL_HRTIM_ADCTRIG_SRC13_TIMDCMP2 | ADC Trigger on Timer D compare 2 |
| LL_HRTIM_ADCTRIG_SRC13_TIMDCMP3 | ADC Trigger on Timer D compare 3 |
| LL_HRTIM_ADCTRIG_SRC13_TIMDCMP4 | ADC Trigger on Timer D compare 4 |
| LL_HRTIM_ADCTRIG_SRC13_TIMDPER  | ADC Trigger on Timer D period    |
| LL_HRTIM_ADCTRIG_SRC13_TIMECMP2 | ADC Trigger on Timer E compare 2 |
| LL_HRTIM_ADCTRIG_SRC13_TIMECMP3 | ADC Trigger on Timer E compare 3 |
| LL_HRTIM_ADCTRIG_SRC13_TIMECMP4 | ADC Trigger on Timer E compare 4 |
| LL_HRTIM_ADCTRIG_SRC13_TIMEPER  | ADC Trigger on Timer E period    |
| <b>ADC TRIGGER 2/4 SOURCE</b>   |                                  |
| LL_HRTIM_ADCTRIG_SRC24_NONE     | No ADC trigger event             |
| LL_HRTIM_ADCTRIG_SRC24_MCMP1    | ADC Trigger on master compare 1  |
| LL_HRTIM_ADCTRIG_SRC24_MCMP2    | ADC Trigger on master compare 2  |
| LL_HRTIM_ADCTRIG_SRC24_MCMP3    | ADC Trigger on master compare 3  |
| LL_HRTIM_ADCTRIG_SRC24_MCMP4    | ADC Trigger on master compare 4  |
| LL_HRTIM_ADCTRIG_SRC24_MPER     | ADC Trigger on master period     |
| LL_HRTIM_ADCTRIG_SRC24_EEV6     | ADC Trigger on external event 6  |
| LL_HRTIM_ADCTRIG_SRC24_EEV7     | ADC Trigger on external event 7  |
| LL_HRTIM_ADCTRIG_SRC24_EEV8     | ADC Trigger on external event 8  |
| LL_HRTIM_ADCTRIG_SRC24_EEV9     | ADC Trigger on external event 9  |
| LL_HRTIM_ADCTRIG_SRC24_EEV10    | ADC Trigger on external event 10 |
| LL_HRTIM_ADCTRIG_SRC24_TIMACMP2 | ADC Trigger on Timer A compare 2 |
| LL_HRTIM_ADCTRIG_SRC24_TIMACMP3 | ADC Trigger on Timer A compare 3 |
| LL_HRTIM_ADCTRIG_SRC24_TIMACMP4 | ADC Trigger on Timer A compare 4 |
| LL_HRTIM_ADCTRIG_SRC24_TIMAPER  | ADC Trigger on Timer A period    |
| LL_HRTIM_ADCTRIG_SRC24_TIMBCMP2 | ADC Trigger on Timer B compare 2 |
| LL_HRTIM_ADCTRIG_SRC24_TIMBCMP3 | ADC Trigger on Timer B compare 3 |
| LL_HRTIM_ADCTRIG_SRC24_TIMBCMP4 | ADC Trigger on Timer B compare 4 |
| LL_HRTIM_ADCTRIG_SRC24_TIMBPER  | ADC Trigger on Timer B period    |
| LL_HRTIM_ADCTRIG_SRC24_TIMCCMP2 | ADC Trigger on Timer C compare 2 |
| LL_HRTIM_ADCTRIG_SRC24_TIMCCMP3 | ADC Trigger on Timer C compare 3 |
| LL_HRTIM_ADCTRIG_SRC24_TIMCCMP4 | ADC Trigger on Timer C compare 4 |

|                                              |                                                                                         |
|----------------------------------------------|-----------------------------------------------------------------------------------------|
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMCPER</code>  | ADC Trigger on Timer C period                                                           |
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMCRST</code>  | ADC Trigger on Timer C reset                                                            |
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMDCMP2</code> | ADC Trigger on Timer D compare 2                                                        |
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMDCMP3</code> | ADC Trigger on Timer D compare 3                                                        |
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMDCMP4</code> | ADC Trigger on Timer D compare 4                                                        |
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMDPER</code>  | ADC Trigger on Timer D period                                                           |
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMDRST</code>  | ADC Trigger on Timer D reset                                                            |
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMECMP2</code> | ADC Trigger on Timer E compare 2                                                        |
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMECMP3</code> | ADC Trigger on Timer E compare 3                                                        |
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMECMP4</code> | ADC Trigger on Timer E compare 4                                                        |
| <code>LL_HRTIM_ADCTRIG_SRC24_TIMERST</code>  | ADC Trigger on Timer E reset                                                            |
| <b>ADC TRIGGER UPDATE</b>                    |                                                                                         |
| <code>LL_HRTIM_ADCTRIG_UPDATE_MASTER</code>  | HRTIM_ADCxR register update is triggered by the Master timer                            |
| <code>LL_HRTIM_ADCTRIG_UPDATE_TIMER_A</code> | HRTIM_ADCxR register update is triggered by the Timer A                                 |
| <code>LL_HRTIM_ADCTRIG_UPDATE_TIMER_B</code> | HRTIM_ADCxR register update is triggered by the Timer B                                 |
| <code>LL_HRTIM_ADCTRIG_UPDATE_TIMER_C</code> | HRTIM_ADCxR register update is triggered by the Timer C                                 |
| <code>LL_HRTIM_ADCTRIG_UPDATE_TIMER_D</code> | HRTIM_ADCxR register update is triggered by the Timer D                                 |
| <code>LL_HRTIM_ADCTRIG_UPDATE_TIMER_E</code> | HRTIM_ADCxR register update is triggered by the Timer E                                 |
| <b>BURST MODE CLOCK SOURCE</b>               |                                                                                         |
| <code>LL_HRTIM_BM_CLKSRC_MASTER</code>       | Master timer counter reset/roll-over is used as clock source for the burst mode counter |
| <code>LL_HRTIM_BM_CLKSRC_TIMER_A</code>      | Timer A counter reset/roll-over is used as clock source for the burst mode counter      |
| <code>LL_HRTIM_BM_CLKSRC_TIMER_B</code>      | Timer B counter reset/roll-over is used as clock source for the burst mode counter      |
| <code>LL_HRTIM_BM_CLKSRC_TIMER_C</code>      | Timer C counter reset/roll-over is used as clock source for the burst mode counter      |
| <code>LL_HRTIM_BM_CLKSRC_TIMER_D</code>      | Timer D counter reset/roll-over is used as clock source for the burst mode counter      |
| <code>LL_HRTIM_BM_CLKSRC_TIMER_E</code>      | Timer E counter reset/roll-over is used as clock source for the burst mode counter      |
| <code>LL_HRTIM_BM_CLKSRC_TIM16_OC</code>     | On-chip Event 1 (BMCIk[1]), acting as a burst mode counter clock                        |
| <code>LL_HRTIM_BM_CLKSRC_TIM17_OC</code>     | On-chip Event 2 (BMCIk[2]), acting as a burst mode counter clock                        |

|                                    |                                                                           |
|------------------------------------|---------------------------------------------------------------------------|
| LL_HRTIM_BM_CLKSRC_TIM7_TRGO       | On-chip Event 3 (BMCIk[3]), acting as a burst mode counter clock          |
| LL_HRTIM_BM_CLKSRC_FHRTIM          | Prescaled fHRTIM clock is used as clock source for the burst mode counter |
| <b>BURST MODE OPERATING MODE</b>   |                                                                           |
| LL_HRTIM_BM_MODE_SINGLESHOT        | Burst mode operates in single shot mode                                   |
| LL_HRTIM_BM_MODE_CONTINOUS         | Burst mode operates in continuous mode                                    |
| <b>BURST MODE PRESCALER</b>        |                                                                           |
| LL_HRTIM_BM_PRESCALER_DIV1         | fBRST = fHRTIM                                                            |
| LL_HRTIM_BM_PRESCALER_DIV2         | fBRST = fHRTIM/2                                                          |
| LL_HRTIM_BM_PRESCALER_DIV4         | fBRST = fHRTIM/4                                                          |
| LL_HRTIM_BM_PRESCALER_DIV8         | fBRST = fHRTIM/8                                                          |
| LL_HRTIM_BM_PRESCALER_DIV16        | fBRST = fHRTIM/16                                                         |
| LL_HRTIM_BM_PRESCALER_DIV32        | fBRST = fHRTIM/32                                                         |
| LL_HRTIM_BM_PRESCALER_DIV64        | fBRST = fHRTIM/64                                                         |
| LL_HRTIM_BM_PRESCALER_DIV128       | fBRST = fHRTIM/128                                                        |
| LL_HRTIM_BM_PRESCALER_DIV256       | fBRST = fHRTIM/256                                                        |
| LL_HRTIM_BM_PRESCALER_DIV512       | fBRST = fHRTIM/512                                                        |
| LL_HRTIM_BM_PRESCALER_DIV1024      | fBRST = fHRTIM/1024                                                       |
| LL_HRTIM_BM_PRESCALER_DIV2048      | fBRST = fHRTIM/2048                                                       |
| LL_HRTIM_BM_PRESCALER_DIV4096      | fBRST = fHRTIM/4096                                                       |
| LL_HRTIM_BM_PRESCALER_DIV8192      | fBRST = fHRTIM/8192                                                       |
| LL_HRTIM_BM_PRESCALER_DIV16384     | fBRST = fHRTIM/16384                                                      |
| LL_HRTIM_BM_PRESCALER_DIV32768     | fBRST = fHRTIM/32768                                                      |
| <b>HRTIM BURST MODE STATUS</b>     |                                                                           |
| LL_HRTIM_BM_STATUS_NORMAL          | Normal operation                                                          |
| LL_HRTIM_BM_STATUS_BURST_ONGOING   | Burst operation on-going                                                  |
| <b>HRTIM BURST MODE TRIGGER</b>    |                                                                           |
| LL_HRTIM_BM_TRIG_NONE              | No trigger                                                                |
| LL_HRTIM_BM_TRIG_MASTER_RESET      | Master timer reset event is starting the burst mode operation             |
| LL_HRTIM_BM_TRIG_MASTER_REPETITION | Master timer repetition event is starting the burst mode operation        |
| LL_HRTIM_BM_TRIG_MASTER_CMP1       | Master timer compare 1 event is starting the burst mode operation         |
| LL_HRTIM_BM_TRIG_MASTER_CMP2       | Master timer compare 2 event is starting the burst mode operation         |
| LL_HRTIM_BM_TRIG_MASTER_CMP3       | Master timer compare 3 event is starting the burst mode operation         |

|                                  |                                                                               |
|----------------------------------|-------------------------------------------------------------------------------|
| LL_HRTIM_BM_TRIG_MASTER_CMP4     | Master timer compare 4 event is starting the burst mode operation             |
| LL_HRTIM_BM_TRIG_TIMA_RESET      | Timer A reset event is starting the burst mode operation                      |
| LL_HRTIM_BM_TRIG_TIMA_REPETITION | Timer A repetition event is starting the burst mode operation                 |
| LL_HRTIM_BM_TRIG_TIMA_CMP1       | Timer A compare 1 event is starting the burst mode operation                  |
| LL_HRTIM_BM_TRIG_TIMA_CMP2       | Timer A compare 2 event is starting the burst mode operation                  |
| LL_HRTIM_BM_TRIG_TIMB_RESET      | Timer B reset event is starting the burst mode operation                      |
| LL_HRTIM_BM_TRIG_TIMB_REPETITION | Timer B repetition event is starting the burst mode operation                 |
| LL_HRTIM_BM_TRIG_TIMB_CMP1       | Timer B compare 1 event is starting the burst mode operation                  |
| LL_HRTIM_BM_TRIG_TIMB_CMP2       | Timer B compare 2 event is starting the burst mode operation                  |
| LL_HRTIM_BM_TRIG_TIMC_RESET      | Timer C reset event is starting the burst mode operation                      |
| LL_HRTIM_BM_TRIG_TIMC_REPETITION | Timer C repetition event is starting the burst mode operation                 |
| LL_HRTIM_BM_TRIG_TIMC_CMP1       | Timer C compare 1 event is starting the burst mode operation                  |
| LL_HRTIM_BM_TRIG_TIMC_CMP2       | Timer C compare 2 event is starting the burst mode operation                  |
| LL_HRTIM_BM_TRIG_TIMD_RESET      | Timer D reset event is starting the burst mode operation                      |
| LL_HRTIM_BM_TRIG_TIMD_REPETITION | Timer D repetition event is starting the burst mode operation                 |
| LL_HRTIM_BM_TRIG_TIMD_CMP1       | Timer D compare 1 event is starting the burst mode operation                  |
| LL_HRTIM_BM_TRIG_TIMD_CMP2       | Timer D compare 2 event is starting the burst mode operation                  |
| LL_HRTIM_BM_TRIG_TIME_RESET      | Timer E reset event is starting the burst mode operation                      |
| LL_HRTIM_BM_TRIG_TIME_REPETITION | Timer E repetition event is starting the burst mode operation                 |
| LL_HRTIM_BM_TRIG_TIME_CMP1       | Timer E compare 1 event is starting the burst mode operation                  |
| LL_HRTIM_BM_TRIG_TIME_CMP2       | Timer E compare 2 event is starting the burst mode operation                  |
| LL_HRTIM_BM_TRIG_TIMA_EVENT7     | Timer A period following an external event 7 (conditioned by TIMA filters) is |

|                               |                                                                                                                                                      |
|-------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_HRTIM_BM_TRIG_TIMD_EVENT8  | starting the burst mode operation<br>Timer D period following an external event 8 (conditioned by TIMD filters) is starting the burst mode operation |
| LL_HRTIM_BM_TRIG_EVENT_7      | External event 7 conditioned by TIMA filters is starting the burst mode operation                                                                    |
| LL_HRTIM_BM_TRIG_EVENT_8      | External event 8 conditioned by TIMD filters is starting the burst mode operation                                                                    |
| LL_HRTIM_BM_TRIG_EVENT_ONCHIP | A rising edge on an on-chip Event (for instance from GP timer or comparator) triggers the burst mode operation                                       |

**BURST DMA**

|                           |                                                    |
|---------------------------|----------------------------------------------------|
| LL_HRTIM_BURSTDMA_NONE    | No register is updated by Burst DMA accesses       |
| LL_HRTIM_BURSTDMA_MCR     | MCR register is updated by Burst DMA accesses      |
| LL_HRTIM_BURSTDMA_MICR    | MICR register is updated by Burst DMA accesses     |
| LL_HRTIM_BURSTDMA_MDIER   | MDIER register is updated by Burst DMA accesses    |
| LL_HRTIM_BURSTDMA_MCNT    | MCNTR register is updated by Burst DMA accesses    |
| LL_HRTIM_BURSTDMA_MPFR    | MPFR register is updated by Burst DMA accesses     |
| LL_HRTIM_BURSTDMA_MREP    | MREPR register is updated by Burst DMA accesses    |
| LL_HRTIM_BURSTDMA_MCMP1   | MCMP1R register is updated by Burst DMA accesses   |
| LL_HRTIM_BURSTDMA_MCMP2   | MCMP2R register is updated by Burst DMA accesses   |
| LL_HRTIM_BURSTDMA_MCMP3   | MCMP3R register is updated by Burst DMA accesses   |
| LL_HRTIM_BURSTDMA_MCMP4   | MCMP4R register is updated by Burst DMA accesses   |
| LL_HRTIM_BURSTDMA_TIMMCR  | TIMxCR register is updated by Burst DMA accesses   |
| LL_HRTIM_BURSTDMA_TIMICR  | TIMxICR register is updated by Burst DMA accesses  |
| LL_HRTIM_BURSTDMA_TIMDIER | TIMxDIER register is updated by Burst DMA accesses |
| LL_HRTIM_BURSTDMA_TIMCNT  | CNTxCR register is updated by Burst DMA accesses   |
| LL_HRTIM_BURSTDMA_TIMPER  | PERxR register is updated by Burst DMA accesses    |
| LL_HRTIM_BURSTDMA_TIMREP  | REPxR register is updated by Burst DMA accesses    |
| LL_HRTIM_BURSTDMA_TIMCMP1 | CMP1xR register is updated by Burst DMA accesses   |

|                                   |                                                                   |
|-----------------------------------|-------------------------------------------------------------------|
| LL_HRTIM_BURSTDMA_TIMCMP2         | CMP2xR register is updated by Burst DMA accesses                  |
| LL_HRTIM_BURSTDMA_TIMCMP3         | CMP3xR register is updated by Burst DMA accesses                  |
| LL_HRTIM_BURSTDMA_TIMCMP4         | CMP4xR register is updated by Burst DMA accesses                  |
| LL_HRTIM_BURSTDMA_TIMDTR          | DTxR register is updated by Burst DMA accesses                    |
| LL_HRTIM_BURSTDMA_TIMSET1R        | SET1R register is updated by Burst DMA accesses                   |
| LL_HRTIM_BURSTDMA_TIMRST1R        | RST1R register is updated by Burst DMA accesses                   |
| LL_HRTIM_BURSTDMA_TIMSET2R        | SET2R register is updated by Burst DMA accesses                   |
| LL_HRTIM_BURSTDMA_TIMRST2R        | RST1R register is updated by Burst DMA accesses                   |
| LL_HRTIM_BURSTDMA_TIMEEFR1        | EEFxR1 register is updated by Burst DMA accesses                  |
| LL_HRTIM_BURSTDMA_TIMEEFR2        | EEFxR2 register is updated by Burst DMA accesses                  |
| LL_HRTIM_BURSTDMA_TIMRSTR         | RSTxR register is updated by Burst DMA accesses                   |
| LL_HRTIM_BURSTDMA_TIMCHPR         | CHPxR register is updated by Burst DMA accesses                   |
| LL_HRTIM_BURSTDMA_TIMOUTR         | OUTxR register is updated by Burst DMA accesses                   |
| LL_HRTIM_BURSTDMA_TIMFLTR         | FLTxR register is updated by Burst DMA accesses                   |
| <b>BURST MODE</b>                 |                                                                   |
| LL_HRTIM_BURSTMODE_MAINTAINCLOCK  | Timer counter clock is maintained and the timer operates normally |
| LL_HRTIM_BURSTMODE_RESETCOUNTER   | Timer counter clock is stopped and the counter is reset           |
| <b>DLL CALIBRATION RATE</b>       |                                                                   |
| LL_HRTIM_DLLCALIBRATION_RATE_7300 | Periodic DLL calibration: $T = 1048576 * tHRTIM$ (7.3 ms)         |
| LL_HRTIM_DLLCALIBRATION_RATE_910  | Periodic DLL calibration: $T = 131072 * tHRTIM$ (910 ms)          |
| LL_HRTIM_DLLCALIBRATION_RATE_114  | Periodic DLL calibration: $T = 16384 * tHRTIM$ (114 ms)           |
| LL_HRTIM_DLLCALIBRATION_RATE_14   | Periodic DLL calibration: $T = 2048 * tHRTIM$ (14 ms)             |
| <b>CAPTURE TRIGGER</b>            |                                                                   |
| LL_HRTIM_CAPTURETRIG_NONE         | Capture trigger is disabled                                       |
| LL_HRTIM_CAPTURETRIG_UPDATE       | The update event triggers the Capture                             |
| LL_HRTIM_CAPTURETRIG_EEV_1        | The External event 1 triggers the Capture                         |
| LL_HRTIM_CAPTURETRIG_EEV_2        | The External event 2 triggers the Capture                         |
| LL_HRTIM_CAPTURETRIG_EEV_3        | The External event 3 triggers the Capture                         |
| LL_HRTIM_CAPTURETRIG_EEV_4        | The External event 4 triggers the Capture                         |

|                                |                                                                  |
|--------------------------------|------------------------------------------------------------------|
| LL_HRTIM_CAPTURETRIG_EEV_5     | The External event 5 triggers the Capture                        |
| LL_HRTIM_CAPTURETRIG_EEV_6     | The External event 6 triggers the Capture                        |
| LL_HRTIM_CAPTURETRIG_EEV_7     | The External event 7 triggers the Capture                        |
| LL_HRTIM_CAPTURETRIG_EEV_8     | The External event 8 triggers the Capture                        |
| LL_HRTIM_CAPTURETRIG_EEV_9     | The External event 9 triggers the Capture                        |
| LL_HRTIM_CAPTURETRIG_EEV_10    | The External event 10 triggers the Capture                       |
| LL_HRTIM_CAPTURETRIG_TA1_SET   | Capture is triggered by TA1 output inactive to active transition |
| LL_HRTIM_CAPTURETRIG_TA1_RESET | Capture is triggered by TA1 output active to inactive transition |
| LL_HRTIM_CAPTURETRIG_TIMA_CMP1 | Timer A Compare 1 triggers Capture                               |
| LL_HRTIM_CAPTURETRIG_TIMA_CMP2 | Timer A Compare 2 triggers Capture                               |
| LL_HRTIM_CAPTURETRIG_TB1_SET   | Capture is triggered by TB1 output inactive to active transition |
| LL_HRTIM_CAPTURETRIG_TB1_RESET | Capture is triggered by TB1 output active to inactive transition |
| LL_HRTIM_CAPTURETRIG_TIMB_CMP1 | Timer B Compare 1 triggers Capture                               |
| LL_HRTIM_CAPTURETRIG_TIMB_CMP2 | Timer B Compare 2 triggers Capture                               |
| LL_HRTIM_CAPTURETRIG_TC1_SET   | Capture is triggered by TC1 output inactive to active transition |
| LL_HRTIM_CAPTURETRIG_TC1_RESET | Capture is triggered by TC1 output active to inactive transition |
| LL_HRTIM_CAPTURETRIG_TIMC_CMP1 | Timer C Compare 1 triggers Capture                               |
| LL_HRTIM_CAPTURETRIG_TIMC_CMP2 | Timer C Compare 2 triggers Capture                               |
| LL_HRTIM_CAPTURETRIG_TD1_SET   | Capture is triggered by TD1 output inactive to active transition |
| LL_HRTIM_CAPTURETRIG_TD1_RESET | Capture is triggered by TD1 output active to inactive transition |
| LL_HRTIM_CAPTURETRIG_TIMD_CMP1 | Timer D Compare 1 triggers Capture                               |
| LL_HRTIM_CAPTURETRIG_TIMD_CMP2 | Timer D Compare 2 triggers Capture                               |
| LL_HRTIM_CAPTURETRIG_TE1_SET   | Capture is triggered by TE1 output inactive to active transition |
| LL_HRTIM_CAPTURETRIG_TE1_RESET | Capture is triggered by TE1 output active to inactive transition |
| LL_HRTIM_CAPTURETRIG_TIME_CMP1 | Timer E Compare 1 triggers Capture                               |
| LL_HRTIM_CAPTURETRIG_TIME_CMP2 | Timer E Compare 2 triggers Capture                               |
| <b>CAPTURE UNIT ID</b>         |                                                                  |
| LL_HRTIM_CAPTUREUNIT_1         | Capture unit 1 identifier                                        |
| LL_HRTIM_CAPTUREUNIT_2         | Capture unit 2 identifier                                        |

***CHOPPER MODE DUTY CYCLE***

|                            |                                            |
|----------------------------|--------------------------------------------|
| LL_HRTIM_CHP_DUTYCYCLE_0   | Only 1st pulse is present                  |
| LL_HRTIM_CHP_DUTYCYCLE_125 | Duty cycle of the carrier signal is 12.5 % |
| LL_HRTIM_CHP_DUTYCYCLE_250 | Duty cycle of the carrier signal is 25 %   |
| LL_HRTIM_CHP_DUTYCYCLE_375 | Duty cycle of the carrier signal is 37.5 % |
| LL_HRTIM_CHP_DUTYCYCLE_500 | Duty cycle of the carrier signal is 50 %   |
| LL_HRTIM_CHP_DUTYCYCLE_625 | Duty cycle of the carrier signal is 62.5 % |
| LL_HRTIM_CHP_DUTYCYCLE_750 | Duty cycle of the carrier signal is 75 %   |
| LL_HRTIM_CHP_DUTYCYCLE_875 | Duty cycle of the carrier signal is 87.5 % |

***CHOPPER MODE PRESCALER***

|                               |                                |
|-------------------------------|--------------------------------|
| LL_HRTIM_CHP_PRESCALER_DIV16  | $f_{CHPFRQ} = f_{HRTIM} / 16$  |
| LL_HRTIM_CHP_PRESCALER_DIV32  | $f_{CHPFRQ} = f_{HRTIM} / 32$  |
| LL_HRTIM_CHP_PRESCALER_DIV48  | $f_{CHPFRQ} = f_{HRTIM} / 48$  |
| LL_HRTIM_CHP_PRESCALER_DIV64  | $f_{CHPFRQ} = f_{HRTIM} / 64$  |
| LL_HRTIM_CHP_PRESCALER_DIV80  | $f_{CHPFRQ} = f_{HRTIM} / 80$  |
| LL_HRTIM_CHP_PRESCALER_DIV96  | $f_{CHPFRQ} = f_{HRTIM} / 96$  |
| LL_HRTIM_CHP_PRESCALER_DIV112 | $f_{CHPFRQ} = f_{HRTIM} / 112$ |
| LL_HRTIM_CHP_PRESCALER_DIV128 | $f_{CHPFRQ} = f_{HRTIM} / 128$ |
| LL_HRTIM_CHP_PRESCALER_DIV144 | $f_{CHPFRQ} = f_{HRTIM} / 144$ |
| LL_HRTIM_CHP_PRESCALER_DIV160 | $f_{CHPFRQ} = f_{HRTIM} / 160$ |
| LL_HRTIM_CHP_PRESCALER_DIV176 | $f_{CHPFRQ} = f_{HRTIM} / 176$ |
| LL_HRTIM_CHP_PRESCALER_DIV192 | $f_{CHPFRQ} = f_{HRTIM} / 192$ |
| LL_HRTIM_CHP_PRESCALER_DIV208 | $f_{CHPFRQ} = f_{HRTIM} / 208$ |
| LL_HRTIM_CHP_PRESCALER_DIV224 | $f_{CHPFRQ} = f_{HRTIM} / 224$ |
| LL_HRTIM_CHP_PRESCALER_DIV240 | $f_{CHPFRQ} = f_{HRTIM} / 240$ |
| LL_HRTIM_CHP_PRESCALER_DIV256 | $f_{CHPFRQ} = f_{HRTIM} / 256$ |

***CHOPPER MODE PULSE WIDTH***

|                             |                                   |
|-----------------------------|-----------------------------------|
| LL_HRTIM_CHP_PULSEWIDTH_16  | $t_{STPW} = t_{HRTIM} \times 16$  |
| LL_HRTIM_CHP_PULSEWIDTH_32  | $t_{STPW} = t_{HRTIM} \times 32$  |
| LL_HRTIM_CHP_PULSEWIDTH_48  | $t_{STPW} = t_{HRTIM} \times 48$  |
| LL_HRTIM_CHP_PULSEWIDTH_64  | $t_{STPW} = t_{HRTIM} \times 64$  |
| LL_HRTIM_CHP_PULSEWIDTH_80  | $t_{STPW} = t_{HRTIM} \times 80$  |
| LL_HRTIM_CHP_PULSEWIDTH_96  | $t_{STPW} = t_{HRTIM} \times 96$  |
| LL_HRTIM_CHP_PULSEWIDTH_112 | $t_{STPW} = t_{HRTIM} \times 112$ |
| LL_HRTIM_CHP_PULSEWIDTH_128 | $t_{STPW} = t_{HRTIM} \times 128$ |
| LL_HRTIM_CHP_PULSEWIDTH_144 | $t_{STPW} = t_{HRTIM} \times 144$ |

|                             |                      |
|-----------------------------|----------------------|
| LL_HRTIM_CHP_PULSEWIDTH_160 | tSTPW = tHRTIM x 160 |
| LL_HRTIM_CHP_PULSEWIDTH_176 | tSTPW = tHRTIM x 176 |
| LL_HRTIM_CHP_PULSEWIDTH_192 | tSTPW = tHRTIM x 192 |
| LL_HRTIM_CHP_PULSEWIDTH_208 | tSTPW = tHRTIM x 208 |
| LL_HRTIM_CHP_PULSEWIDTH_224 | tSTPW = tHRTIM x 224 |
| LL_HRTIM_CHP_PULSEWIDTH_240 | tSTPW = tHRTIM x 240 |
| LL_HRTIM_CHP_PULSEWIDTH_256 | tSTPW = tHRTIM x 256 |

**COMPARE MODE**

|                                      |                                                                                                                    |
|--------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| LL_HRTIM_COMPAREMODE_REGULAR         | standard compare mode                                                                                              |
| LL_HRTIM_COMPAREMODE_DELAY_NOTIMEOUT | Compare event generated only if a capture has occurred                                                             |
| LL_HRTIM_COMPAREMODE_DELAY_CMP1      | Compare event generated if a capture has occurred or after a Compare 1 match (timeout if capture event is missing) |
| LL_HRTIM_COMPAREMODE_DELAY_CMP3      | Compare event generated if a capture has occurred or after a Compare 3 match (timeout if capture event is missing) |

**COMPARE UNIT ID**

|                        |                           |
|------------------------|---------------------------|
| LL_HRTIM_COMPAREUNIT_2 | Compare unit 2 identifier |
| LL_HRTIM_COMPAREUNIT_4 | Compare unit 4 identifier |

**CURRENT PUSH-PULL STATUS**

|                          |                                                         |
|--------------------------|---------------------------------------------------------|
| LL_HRTIM_CPPSTAT_OUTPUT1 | Signal applied on output 1 and output 2 forced inactive |
| LL_HRTIM_CPPSTAT_OUTPUT2 | Signal applied on output 2 and output 1 forced inactive |

**CROSSBAR INPUT**

|                           |                                                                                               |
|---------------------------|-----------------------------------------------------------------------------------------------|
| LL_HRTIM_CROSSBAR_NONE    | Reset the output set crossbar                                                                 |
| LL_HRTIM_CROSSBAR_RESYNC  | Timer reset event coming solely from software or SYNC input forces an output level transition |
| LL_HRTIM_CROSSBAR_TIMPER  | Timer period event forces an output level transition                                          |
| LL_HRTIM_CROSSBAR_TIMCMP1 | Timer compare 1 event forces an output level transition                                       |
| LL_HRTIM_CROSSBAR_TIMCMP2 | Timer compare 2 event forces an output level transition                                       |
| LL_HRTIM_CROSSBAR_TIMCMP3 | Timer compare 3 event forces an output level transition                                       |
| LL_HRTIM_CROSSBAR_TIMCMP4 | Timer compare 4 event forces an output level transition                                       |

|                              |                                                                 |
|------------------------------|-----------------------------------------------------------------|
| LL_HRTIM_CROSSBAR_MASTERPER  | The master timer period event forces an output level transision |
| LL_HRTIM_CROSSBAR_MASTERCMP1 | Master Timer compare 1 event forces an output level transision  |
| LL_HRTIM_CROSSBAR_MASTERCMP2 | Master Timer compare 2 event forces an output level transision  |
| LL_HRTIM_CROSSBAR_MASTERCMP3 | Master Timer compare 3 event forces an output level transision  |
| LL_HRTIM_CROSSBAR_MASTERCMP4 | Master Timer compare 4 event forces an output level transision  |
| LL_HRTIM_CROSSBAR_TIMEV_1    | Timer event 1 forces an output level transision                 |
| LL_HRTIM_CROSSBAR_TIMEV_2    | Timer event 2 forces an output level transision                 |
| LL_HRTIM_CROSSBAR_TIMEV_3    | Timer event 3 forces an output level transision                 |
| LL_HRTIM_CROSSBAR_TIMEV_4    | Timer event 4 forces an output level transision                 |
| LL_HRTIM_CROSSBAR_TIMEV_5    | Timer event 5 forces an output level transision                 |
| LL_HRTIM_CROSSBAR_TIMEV_6    | Timer event 6 forces an output level transision                 |
| LL_HRTIM_CROSSBAR_TIMEV_7    | Timer event 7 forces an output level transision                 |
| LL_HRTIM_CROSSBAR_TIMEV_8    | Timer event 8 forces an output level transision                 |
| LL_HRTIM_CROSSBAR_TIMEV_9    | Timer event 9 forces an output level transision                 |
| LL_HRTIM_CROSSBAR_EEV_1      | External event 1 forces an output level transision              |
| LL_HRTIM_CROSSBAR_EEV_2      | External event 2 forces an output level transision              |
| LL_HRTIM_CROSSBAR_EEV_3      | External event 3 forces an output level transision              |
| LL_HRTIM_CROSSBAR_EEV_4      | External event 4 forces an output level transision              |
| LL_HRTIM_CROSSBAR_EEV_5      | External event 5 forces an output level transision              |
| LL_HRTIM_CROSSBAR_EEV_6      | External event 6 forces an output level transision              |
| LL_HRTIM_CROSSBAR_EEV_7      | External event 7 forces an output level transision              |
| LL_HRTIM_CROSSBAR_EEV_8      | External event 8 forces an output level transision              |
| LL_HRTIM_CROSSBAR_EEV_9      | External event 9 forces an output level transision              |
| LL_HRTIM_CROSSBAR_EEV_10     | External event 10 forces an output level transision             |
| LL_HRTIM_CROSSBAR_UPDATE     | Timer register update event forces an output level transision   |

**DAC TRIGGER**

|                               |                                                                             |
|-------------------------------|-----------------------------------------------------------------------------|
| LL_HRTIM_DACTRIG_NONE         | No DAC synchronization event generated                                      |
| LL_HRTIM_DACTRIG_DACTRIGOUT_1 | DAC synchronization event generated on DACTrigOut1 output upon timer update |
| LL_HRTIM_DACTRIG_DACTRIGOUT_2 | DAC synchronization event generated on DACTrigOut2 output upon timer update |
| LL_HRTIM_DACTRIG_DACTRIGOUT_3 | DAC synchronization event generated on DACTrigOut3 output upon timer update |

**DLL CALIBRATION MODE**

|                                         |                                       |
|-----------------------------------------|---------------------------------------|
| LL_HRTIM_DLLCALIBRATION_MODE_SINGLESHOT | Calibration is performed only once    |
| LL_HRTIM_DLLCALIBRATION_MODE_CONTINUOUS | Calibration is performed periodically |

**DELAYED PROTECTION (DLYPRT) MODE**

|                                  |                                                                        |
|----------------------------------|------------------------------------------------------------------------|
| LL_HRTIM_DLYPRT_DELAYAYOUT1_EEV6 | Timers A, B, C: Output 1 delayed Idle on external Event 6              |
| LL_HRTIM_DLYPRT_DELAYAYOUT2_EEV6 | Timers A, B, C: Output 2 delayed Idle on external Event 6              |
| LL_HRTIM_DLYPRT_DELAYBOTH_EEV6   | Timers A, B, C: Output 1 and output 2 delayed Idle on external Event 6 |
| LL_HRTIM_DLYPRT_BALANCED_EEV6    | Timers A, B, C: Balanced Idle on external Event 6                      |
| LL_HRTIM_DLYPRT_DELAYAYOUT1_EEV7 | Timers A, B, C: Output 1 delayed Idle on external Event 7              |
| LL_HRTIM_DLYPRT_DELAYAYOUT2_EEV7 | Timers A, B, C: Output 2 delayed Idle on external Event 7              |
| LL_HRTIM_DLYPRT_DELAYBOTH_EEV7   | Timers A, B, C: Output 1 and output2 delayed Idle on external Event 7  |
| LL_HRTIM_DLYPRT_BALANCED_EEV7    | Timers A, B, C: Balanced Idle on external Event 7                      |
| LL_HRTIM_DLYPRT_DELAYAYOUT1_EEV8 | Timers D, E: Output 1 delayed Idle on external Event 8                 |
| LL_HRTIM_DLYPRT_DELAYAYOUT2_EEV8 | Timers D, E: Output 2 delayed Idle on external Event 8                 |
| LL_HRTIM_DLYPRT_DELAYBOTH_EEV8   | Timers D, E: Output 1 and output 2 delayed Idle on external Event 8    |
| LL_HRTIM_DLYPRT_BALANCED_EEV8    | Timers D, E: Balanced Idle on external Event 8                         |
| LL_HRTIM_DLYPRT_DELAYAYOUT1_EEV9 | Timers D, E: Output 1 delayed Idle on external Event 9                 |
| LL_HRTIM_DLYPRT_DELAYAYOUT2_EEV9 | Timers D, E: Output 2 delayed Idle on external Event 9                 |
| LL_HRTIM_DLYPRT_DELAYBOTH_EEV9   | Timers D, E: Output 1 and output2 delayed                              |

Idle on external Event 9

`LL_HRTIM_DLYPRT_BALANCED_EEV9` Timers D, E: Balanced Idle on external Event 9

#### **DEADTIME FALLING SIGN**

`LL_HRTIM_DT_FALLING_POSITIVE` Positive deadtime on falling edge

`LL_HRTIM_DT_FALLING_NEGATIVE` Negative deadtime on falling edge

#### **DEADTIME PRESCALER**

`LL_HRTIM_DT_PRESCALER_MUL8`  $fDTG = fHRTIM * 8$

`LL_HRTIM_DT_PRESCALER_MUL4`  $fDTG = fHRTIM * 4$

`LL_HRTIM_DT_PRESCALER_MUL2`  $fDTG = fHRTIM * 2$

`LL_HRTIM_DT_PRESCALER_DIV1`  $fDTG = fHRTIM$

`LL_HRTIM_DT_PRESCALER_DIV2`  $fDTG = fHRTIM / 2$

`LL_HRTIM_DT_PRESCALER_DIV4`  $fDTG = fHRTIM / 4$

`LL_HRTIM_DT_PRESCALER_DIV8`  $fDTG = fHRTIM / 8$

`LL_HRTIM_DT_PRESCALER_DIV16`  $fDTG = fHRTIM / 16$

#### **DEADTIME RISING SIGN**

`LL_HRTIM_DT_RISING_POSITIVE` Positive deadtime on rising edge

`LL_HRTIM_DT_RISING_NEGATIVE` Negative deadtime on rising edge

#### **EXTERNAL EVENT FAST MODE**

`LL_HRTIM_EE_FASTMODE_DISABLE` External Event is re-synchronized by the HRTIM logic before acting on outputs

`LL_HRTIM_EE_FASTMODE_ENABLE` External Event is acting asynchronously on outputs (low latency mode)

#### **EXTERNAL EVENT DIGITAL FILTER**

`LL_HRTIM_EE_FILTER_NONE` Filter disabled

`LL_HRTIM_EE_FILTER_1`  $fSAMPLING = fHRTIM, N=2$

`LL_HRTIM_EE_FILTER_2`  $fSAMPLING = fHRTIM, N=4$

`LL_HRTIM_EE_FILTER_3`  $fSAMPLING = fHRTIM, N=8$

`LL_HRTIM_EE_FILTER_4`  $fSAMPLING = fEEVS/2, N=6$

`LL_HRTIM_EE_FILTER_5`  $fSAMPLING = fEEVS/2, N=8$

`LL_HRTIM_EE_FILTER_6`  $fSAMPLING = fEEVS/4, N=6$

`LL_HRTIM_EE_FILTER_7`  $fSAMPLING = fEEVS/4, N=8$

`LL_HRTIM_EE_FILTER_8`  $fSAMPLING = fEEVS/8, N=6$

`LL_HRTIM_EE_FILTER_9`  $fSAMPLING = fEEVS/8, N=8$

`LL_HRTIM_EE_FILTER_10`  $fSAMPLING = fEEVS/16, N=5$

`LL_HRTIM_EE_FILTER_11`  $fSAMPLING = fEEVS/16, N=6$

`LL_HRTIM_EE_FILTER_12`  $fSAMPLING = fEEVS/16, N=8$

|                       |                           |
|-----------------------|---------------------------|
| LL_HRTIM_EE_FILTER_13 | fSAMPLING = fEEVS/32, N=5 |
| LL_HRTIM_EE_FILTER_14 | fSAMPLING = fEEVS/32, N=6 |
| LL_HRTIM_EE_FILTER_15 | fSAMPLING = fEEVS/32, N=8 |

**EXTERNAL EVENT POLARITY**

|                           |                               |
|---------------------------|-------------------------------|
| LL_HRTIM_EE_POLARITY_HIGH | External event is active high |
| LL_HRTIM_EE_POLARITY_LOW  | External event is active low  |

**EXTERNAL EVENT PRESCALER**

|                            |                    |
|----------------------------|--------------------|
| LL_HRTIM_EE_PRESCALER_DIV1 | fEEVS = fHRTIM     |
| LL_HRTIM_EE_PRESCALER_DIV2 | fEEVS = fHRTIM / 2 |
| LL_HRTIM_EE_PRESCALER_DIV4 | fEEVS = fHRTIM / 4 |
| LL_HRTIM_EE_PRESCALER_DIV8 | fEEVS = fHRTIM / 8 |

**EXTERNAL EVENT SENSITIVITY**

|                                     |                                                      |
|-------------------------------------|------------------------------------------------------|
| LL_HRTIM_EE_SENSITIVITY_LEVEL       | External event is active on level                    |
| LL_HRTIM_EE_SENSITIVITY_RISINGEDGE  | External event is active on Rising edge              |
| LL_HRTIM_EE_SENSITIVITY_FALLINGEDGE | External event is active on Falling edge             |
| LL_HRTIM_EE_SENSITIVITY_BOTHEDGES   | External event is active on Rising and Falling edges |

**EXTERNAL EVENT SOURCE**

|                   |                                   |
|-------------------|-----------------------------------|
| LL_HRTIM_EE_SRC_1 | External event source 1 (EExSrc1) |
| LL_HRTIM_EE_SRC_2 | External event source 2 (EExSrc2) |
| LL_HRTIM_EE_SRC_3 | External event source 3 (EExSrc3) |
| LL_HRTIM_EE_SRC_4 | External event source 4 (EExSrc4) |

**EXTERNAL EVENT ID**

|                   |                                      |
|-------------------|--------------------------------------|
| LL_HRTIM_EVENT_1  | External event channel 1 identifier  |
| LL_HRTIM_EVENT_2  | External event channel 2 identifier  |
| LL_HRTIM_EVENT_3  | External event channel 3 identifier  |
| LL_HRTIM_EVENT_4  | External event channel 4 identifier  |
| LL_HRTIM_EVENT_5  | External event channel 5 identifier  |
| LL_HRTIM_EVENT_6  | External event channel 6 identifier  |
| LL_HRTIM_EVENT_7  | External event channel 7 identifier  |
| LL_HRTIM_EVENT_8  | External event channel 8 identifier  |
| LL_HRTIM_EVENT_9  | External event channel 9 identifier  |
| LL_HRTIM_EVENT_10 | External event channel 10 identifier |

**FAULT ID**

|                  |                            |
|------------------|----------------------------|
| LL_HRTIM_FAULT_1 | Fault channel 1 identifier |
| LL_HRTIM_FAULT_2 | Fault channel 2 identifier |
| LL_HRTIM_FAULT_3 | Fault channel 3 identifier |

`LL_HRTIM_FAULT_4` Fault channel 4 identifier

`LL_HRTIM_FAULT_5` Fault channel 5 identifier

#### **FAULT DIGITAL FILTER**

`LL_HRTIM_FLT_FILTER_NONE` Filter disabled

`LL_HRTIM_FLT_FILTER_1` fSAMPLING= fHRTIM, N=2

`LL_HRTIM_FLT_FILTER_2` fSAMPLING= fHRTIM, N=4

`LL_HRTIM_FLT_FILTER_3` fSAMPLING= fHRTIM, N=8

`LL_HRTIM_FLT_FILTER_4` fSAMPLING= fFLTS/2, N=6

`LL_HRTIM_FLT_FILTER_5` fSAMPLING= fFLTS/2, N=8

`LL_HRTIM_FLT_FILTER_6` fSAMPLING= fFLTS/4, N=6

`LL_HRTIM_FLT_FILTER_7` fSAMPLING= fFLTS/4, N=8

`LL_HRTIM_FLT_FILTER_8` fSAMPLING= fFLTS/8, N=6

`LL_HRTIM_FLT_FILTER_9` fSAMPLING= fFLTS/8, N=8

`LL_HRTIM_FLT_FILTER_10` fSAMPLING= fFLTS/16, N=5

`LL_HRTIM_FLT_FILTER_11` fSAMPLING= fFLTS/16, N=6

`LL_HRTIM_FLT_FILTER_12` fSAMPLING= fFLTS/16, N=8

`LL_HRTIM_FLT_FILTER_13` fSAMPLING= fFLTS/32, N=5

`LL_HRTIM_FLT_FILTER_14` fSAMPLING= fFLTS/32, N=6

`LL_HRTIM_FLT_FILTER_15` fSAMPLING= fFLTS/32, N=8

#### **FAULT POLARITY**

`LL_HRTIM_FLT_POLARITY_LOW` Fault input is active low

`LL_HRTIM_FLT_POLARITY_HIGH` Fault input is active high

#### **BURST FAULT PRESCALER**

`LL_HRTIM_FLT_PRESCALER_DIV1` fFLTS = fHRTIM

`LL_HRTIM_FLT_PRESCALER_DIV2` fFLTS = fHRTIM / 2

`LL_HRTIM_FLT_PRESCALER_DIV4` fFLTS = fHRTIM / 4

`LL_HRTIM_FLT_PRESCALER_DIV8` fFLTS = fHRTIM / 8

#### **FAULT SOURCE**

`LL_HRTIM_FLT_SRC_DIGITALINPUT` Fault input is FLT input pin

`LL_HRTIM_FLT_SRC_INTERNAL` Fault input is FLT\_Int signal (e.g. internal comparator)

#### **Get Flags Defines**

`LL_HRTIM_ISR_FLT1`

`LL_HRTIM_ISR_FLT2`

`LL_HRTIM_ISR_FLT3`

`LL_HRTIM_ISR_FLT4`

`LL_HRTIM_ISR_FLT5`

LL\_HRTIM\_ISR\_SYSFLT  
LL\_HRTIM\_ISR\_DLLRDY  
LL\_HRTIM\_ISR\_BMPER  
LL\_HRTIM\_MISR\_MCMP1  
LL\_HRTIM\_MISR\_MCMP2  
LL\_HRTIM\_MISR\_MCMP3  
LL\_HRTIM\_MISR\_MCMP4  
LL\_HRTIM\_MISR\_MREP  
LL\_HRTIM\_MISR\_SYNC  
LL\_HRTIM\_MISR\_MUPD  
LL\_HRTIM\_TIMISR\_CMP1  
LL\_HRTIM\_TIMISR\_CMP2  
LL\_HRTIM\_TIMISR\_CMP3  
LL\_HRTIM\_TIMISR\_CMP4  
LL\_HRTIM\_TIMISR REP  
LL\_HRTIM\_TIMISR\_UPD  
LL\_HRTIM\_TIMISR\_CPT1  
LL\_HRTIM\_TIMISR\_CPT2  
LL\_HRTIM\_TIMISR\_SET1  
LL\_HRTIM\_TIMISR\_RST1  
LL\_HRTIM\_TIMISR\_SET2  
LL\_HRTIM\_TIMISR\_RST2  
LL\_HRTIM\_TIMISR\_RST  
LL\_HRTIM\_TIMISR\_DLYPRT

**IDLE PUSH-PULL STATUS**

|                          |                                                                               |
|--------------------------|-------------------------------------------------------------------------------|
| LL_HRTIM_IPPSTAT_OUTPUT1 | Protection occurred when the output 1 was active and output 2 forced inactive |
| LL_HRTIM_IPPSTAT_OUTPUT2 | Protection occurred when the output 2 was active and output 1 forced inactive |

**IT Defines**

LL\_HRTIM\_IER\_FLT1IE  
LL\_HRTIM\_IER\_FLT2IE  
LL\_HRTIM\_IER\_FLT3IE  
LL\_HRTIM\_IER\_FLT4IE  
LL\_HRTIM\_IER\_FLT5IE  
LL\_HRTIM\_IER\_SYSFLTIE  
LL\_HRTIM\_IER\_DLLRDYIE

LL\_HRTIM\_IER\_BMPERIE  
LL\_HRTIM\_MDIER\_MCMP1IE  
LL\_HRTIM\_MDIER\_MCMP2IE  
LL\_HRTIM\_MDIER\_MCMP3IE  
LL\_HRTIM\_MDIER\_MCMP4IE  
LL\_HRTIM\_MDIER\_MREPIE  
LL\_HRTIM\_MDIER\_SYNCIE  
LL\_HRTIM\_MDIER\_MUPDIE  
LL\_HRTIM\_TIMDIER\_CMP1IE  
LL\_HRTIM\_TIMDIER\_CMP2IE  
LL\_HRTIM\_TIMDIER\_CMP3IE  
LL\_HRTIM\_TIMDIER\_CMP4IE  
LL\_HRTIM\_TIMDIER\_REPIE  
LL\_HRTIM\_TIMDIER\_UPDIE  
LL\_HRTIM\_TIMDIER\_CPT1IE  
LL\_HRTIM\_TIMDIER\_CPT2IE  
LL\_HRTIM\_TIMDIER\_SET1IE  
LL\_HRTIM\_TIMDIER\_RST1IE  
LL\_HRTIM\_TIMDIER\_SET2IE  
LL\_HRTIM\_TIMDIER\_RST2IE  
LL\_HRTIM\_TIMDIER\_RSTIE  
LL\_HRTIM\_TIMDIER\_DLYPRTIE

**COUNTER MODE**

|                             |                                                          |
|-----------------------------|----------------------------------------------------------|
| LL_HRTIM_MODE_CONTINUOUS    | The timer operates in continuous (free-running) mode     |
| LL_HRTIM_MODE_SINGLESHOT    | The timer operates in non retriggerable single-shot mode |
| LL_HRTIM_MODE_RETRIGGERABLE | The timer operates in retriggerable single-shot mode     |

**OUTPUT ID**

|                     |                               |
|---------------------|-------------------------------|
| LL_HRTIM_OUTPUT_TA1 | Timer A - Output 1 identifier |
| LL_HRTIM_OUTPUT_TA2 | Timer A - Output 2 identifier |
| LL_HRTIM_OUTPUT_TB1 | Timer B - Output 1 identifier |
| LL_HRTIM_OUTPUT_TB2 | Timer B - Output 2 identifier |
| LL_HRTIM_OUTPUT_TC1 | Timer C - Output 1 identifier |
| LL_HRTIM_OUTPUT_TC2 | Timer C - Output 2 identifier |
| LL_HRTIM_OUTPUT_TD1 | Timer D - Output 1 identifier |

`LL_HRTIM_OUTPUT_TD2` Timer D - Output 2 identifier

`LL_HRTIM_OUTPUT_TE1` Timer E - Output 1 identifier

`LL_HRTIM_OUTPUT_TE2` Timer E - Output 2 identifier

#### **OUTPUT STATE**

`LL_HRTIM_OUTPUTSTATE_IDLE` Main operating mode, where the output can take the active or inactive level as programmed in the crossbar unit

`LL_HRTIM_OUTPUTSTATE_RUN` Default operating state (e.g. after an HRTIM reset, when the outputs are disabled by software or during a burst mode operation)

`LL_HRTIM_OUTPUTSTATE_FAULT` Safety state, entered in case of a shut-down request on FAULTx inputs

#### **OUTPUT BURST MODE ENTRY MODE**

`LL_HRTIM_OUT_BM_ENTRYMODE_REGULAR` The programmed Idle state is applied immediately to the Output

`LL_HRTIM_OUT_BM_ENTRYMODE_DELAYED` Deadtime is inserted on output before entering the idle mode

#### **OUTPUT CHOPPER MODE**

`LL_HRTIM_OUT_CHOPPERMODE_DISABLED` Output signal is not altered

`LL_HRTIM_OUT_CHOPPERMODE_ENABLED` Output signal is chopped by a carrier signal

#### **OUTPUT FAULT STATE**

`LL_HRTIM_OUT_FAULTSTATE_NO_ACTION` The output is not affected by the fault input

`LL_HRTIM_OUT_FAULTSTATE_ACTIVE` Output at active level when in FAULT state

`LL_HRTIM_OUT_FAULTSTATE_INACTIVE` Output at inactive level when in FAULT state

`LL_HRTIM_OUT_FAULTSTATE_HIGHZ` Output is tri-stated when in FAULT state

#### **OUTPUT IDLE LEVEL**

`LL_HRTIM_OUT_IDLELEVEL_INACTIVE` Output at inactive level when in IDLE state

`LL_HRTIM_OUT_IDLELEVEL_ACTIVE` Output at active level when in IDLE state

#### **OUTPUT IDLE MODE**

`LL_HRTIM_OUT_NO_IDLE` The output is not affected by the burst mode operation

`LL_HRTIM_OUT_IDLE_WHEN_BURST` The output is in idle state when requested by the burst mode controller

#### **OUTPUT LEVEL**

`LL_HRTIM_OUT_LEVEL_INACTIVE` Corresponds to a logic level 0 for a positive polarity (High) and to a logic level 1 for a negative polarity (Low)

|                                             |                                                                                                                    |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| <code>LL_HRTIM_OUT_LEVEL_ACTIVE</code>      | Corresponds to a logic level 1 for a positive polarity (High) and to a logic level 0 for a negative polarity (Low) |
| <b><i>OUPUT POLARITY</i></b>                |                                                                                                                    |
| <code>LL_HRTIM_OUT_POSITIVE_POLARITY</code> | Output is active HIGH                                                                                              |
| <code>LL_HRTIM_OUT_NEGATIVE_POLARITY</code> | Output is active LOW                                                                                               |
| <b><i>PRESCALER RATIO</i></b>               |                                                                                                                    |
| <code>LL_HRTIM_PRESCALERRATIO_MUL32</code>  | fHRCK: fHRTIM x 32 = 4.608 GHz - Resolution: 217 ps - Min PWM frequency: 70.3 kHz (fHRTIM=144MHz)                  |
| <code>LL_HRTIM_PRESCALERRATIO_MUL16</code>  | fHRCK: fHRTIM x 16 = 2.304 GHz - Resolution: 434 ps - Min PWM frequency: 35.1 kHz (fHRTIM=144MHz)                  |
| <code>LL_HRTIM_PRESCALERRATIO_MUL8</code>   | fHRCK: fHRTIM x 8 = 1.152 GHz - Resolution: 868 ps - Min PWM frequency: 17.6 kHz (fHRTIM=144MHz)                   |
| <code>LL_HRTIM_PRESCALERRATIO_MUL4</code>   | fHRCK: fHRTIM x 4 = 576 MHz - Resolution: 1.73 ns - Min PWM frequency: 8.8 kHz (fHRTIM=144MHz)                     |
| <code>LL_HRTIM_PRESCALERRATIO_MUL2</code>   | fHRCK: fHRTIM x 2 = 288 MHz - Resolution: 3.47 ns - Min PWM frequency: 4.4 kHz (fHRTIM=144MHz)                     |
| <code>LL_HRTIM_PRESCALERRATIO_DIV1</code>   | fHRCK: fHRTIM = 144 MHz - Resolution: 6.95 ns - Min PWM frequency: 2.2 kHz (fHRTIM=144MHz)                         |
| <code>LL_HRTIM_PRESCALERRATIO_DIV2</code>   | fHRCK: fHRTIM / 2 = 72 MHz - Resolution: 13.88 ns - Min PWM frequency: 1.1 kHz (fHRTIM=144MHz)                     |
| <code>LL_HRTIM_PRESCALERRATIO_DIV4</code>   | fHRCK: fHRTIM / 4 = 36 MHz - Resolution: 27.7 ns - Min PWM frequency: 550Hz (fHRTIM=144MHz)                        |
| <b><i>RESET TRIGGER</i></b>                 |                                                                                                                    |
| <code>LL_HRTIM_RESETTRIG_NONE</code>        | No counter reset trigger                                                                                           |
| <code>LL_HRTIM_RESETTRIG_UPDATE</code>      | The timer counter is reset upon update event                                                                       |
| <code>LL_HRTIM_RESETTRIG_CMP2</code>        | The timer counter is reset upon Timer Compare 2 event                                                              |
| <code>LL_HRTIM_RESETTRIG_CMP4</code>        | The timer counter is reset upon Timer Compare 4 event                                                              |
| <code>LL_HRTIM_RESETTRIG_MASTER_PER</code>  | The timer counter is reset upon master timer period event                                                          |
| <code>LL_HRTIM_RESETTRIG_MASTER_CMP1</code> | The timer counter is reset upon master timer Compare 1 event                                                       |
| <code>LL_HRTIM_RESETTRIG_MASTER_CMP2</code> | The timer counter is reset upon master timer Compare 2 event                                                       |

|                                |                                                              |
|--------------------------------|--------------------------------------------------------------|
| LL_HRTIM_RESETTRIG_MASTER_CMP3 | The timer counter is reset upon master timer Compare 3 event |
| LL_HRTIM_RESETTRIG_MASTER_CMP4 | The timer counter is reset upon master timer Compare 4 event |
| LL_HRTIM_RESETTRIG_EEV_1       | The timer counter is reset upon external event 1             |
| LL_HRTIM_RESETTRIG_EEV_2       | The timer counter is reset upon external event 2             |
| LL_HRTIM_RESETTRIG_EEV_3       | The timer counter is reset upon external event 3             |
| LL_HRTIM_RESETTRIG_EEV_4       | The timer counter is reset upon external event 4             |
| LL_HRTIM_RESETTRIG_EEV_5       | The timer counter is reset upon external event 5             |
| LL_HRTIM_RESETTRIG_EEV_6       | The timer counter is reset upon external event 6             |
| LL_HRTIM_RESETTRIG_EEV_7       | The timer counter is reset upon external event 7             |
| LL_HRTIM_RESETTRIG_EEV_8       | The timer counter is reset upon external event 8             |
| LL_HRTIM_RESETTRIG_EEV_9       | The timer counter is reset upon external event 9             |
| LL_HRTIM_RESETTRIG_EEV_10      | The timer counter is reset upon external event 10            |
| LL_HRTIM_RESETTRIG_OTHER1_CMP1 | The timer counter is reset upon other timer Compare 1 event  |
| LL_HRTIM_RESETTRIG_OTHER1_CMP2 | The timer counter is reset upon other timer Compare 2 event  |
| LL_HRTIM_RESETTRIG_OTHER1_CMP4 | The timer counter is reset upon other timer Compare 4 event  |
| LL_HRTIM_RESETTRIG_OTHER2_CMP1 | The timer counter is reset upon other timer Compare 1 event  |
| LL_HRTIM_RESETTRIG_OTHER2_CMP2 | The timer counter is reset upon other timer Compare 2 event  |
| LL_HRTIM_RESETTRIG_OTHER2_CMP4 | The timer counter is reset upon other timer Compare 4 event  |
| LL_HRTIM_RESETTRIG_OTHER3_CMP1 | The timer counter is reset upon other timer Compare 1 event  |
| LL_HRTIM_RESETTRIG_OTHER3_CMP2 | The timer counter is reset upon other timer Compare 2 event  |
| LL_HRTIM_RESETTRIG_OTHER3_CMP4 | The timer counter is reset upon other timer Compare 4 event  |
| LL_HRTIM_RESETTRIG_OTHER4_CMP1 | The timer counter is reset upon other timer Compare 1 event  |

`LL_HRTIM_RESETTRIG_OTHER4_CMP2` The timer counter is reset upon other timer Compare 2 event

`LL_HRTIM_RESETTRIG_OTHER4_CMP4` The timer counter is reset upon other timer Compare 4 event

#### **SYNCHRONIZATION INPUT SOURCE**

`LL_HRTIM_SYNCIN_SRC_NONE` HRTIM is not synchronized and runs in standalone mode

`LL_HRTIM_SYNCIN_SRC_TIM_EVENT` The HRTIM is synchronized with the on-chip timer

`LL_HRTIM_SYNCIN_SRC_EXTERNAL_EVENT` A positive pulse on SYNCIN input triggers the HRTIM

#### **SYNCHRONIZATION OUTPUT POLARITY**

`LL_HRTIM_SYNCOUT_DISABLED` Synchronization output event is disabled

`LL_HRTIM_SYNCOUT_POSITIVE_PULSE` SCOUT pin has a low idle level and issues a positive pulse of 16 fHRTIM clock cycles length for the synchronization

`LL_HRTIM_SYNCOUT_NEGATIVE_PULSE` SCOUT pin has a high idle level and issues a negative pulse of 16 fHRTIM clock cycles length for the synchronization

#### **SYNCHRONIZATION OUTPUT SOURCE**

`LL_HRTIM_SYNCOUT_SRC_MASTER_START` A pulse is sent on the SYNCOUT output upon master timer start event

`LL_HRTIM_SYNCOUT_SRC_MASTER_CMP1` A pulse is sent on the SYNCOUT output upon master timer compare 1 event

`LL_HRTIM_SYNCOUT_SRC_TIMA_START` A pulse is sent on the SYNCOUT output upon timer A start or reset events

`LL_HRTIM_SYNCOUT_SRC_TIMA_CMP1` A pulse is sent on the SYNCOUT output upon timer A compare 1 event

#### **TIMER ID**

`LL_HRTIM_TIMER_MASTER` Master timer identifier

`LL_HRTIM_TIMER_A` Timer A identifier

`LL_HRTIM_TIMER_B` Timer B identifier

`LL_HRTIM_TIMER_C` Timer C identifier

`LL_HRTIM_TIMER_D` Timer D identifier

`LL_HRTIM_TIMER_E` Timer E identifier

#### **TIMER EXTERNAL EVENT FILTER**

`LL_HRTIM_EEFLTR_NONE`

`LL_HRTIM_EEFLTR_BLANKINGCMP1` Blanking from counter reset/roll-over to Compare 1

`LL_HRTIM_EEFLTR_BLANKINGCMP2` Blanking from counter reset/roll-over to Compare 2

|                                            |                                                     |
|--------------------------------------------|-----------------------------------------------------|
| <code>LL_HRTIM_EEFLTR_BLANKINGCMP3</code>  | Blanking from counter reset/roll-over to Compare 3  |
| <code>LL_HRTIM_EEFLTR_BLANKINGCMP4</code>  | Blanking from counter reset/roll-over to Compare 4  |
| <code>LL_HRTIM_EEFLTR_BLANKINGFLTR1</code> | Blanking from another timing unit: TIMFLTR1 source  |
| <code>LL_HRTIM_EEFLTR_BLANKINGFLTR2</code> | Blanking from another timing unit: TIMFLTR2 source  |
| <code>LL_HRTIM_EEFLTR_BLANKINGFLTR3</code> | Blanking from another timing unit: TIMFLTR3 source  |
| <code>LL_HRTIM_EEFLTR_BLANKINGFLTR4</code> | Blanking from another timing unit: TIMFLTR4 source  |
| <code>LL_HRTIM_EEFLTR_BLANKINGFLTR5</code> | Blanking from another timing unit: TIMFLTR5 source  |
| <code>LL_HRTIM_EEFLTR_BLANKINGFLTR6</code> | Blanking from another timing unit: TIMFLTR6 source  |
| <code>LL_HRTIM_EEFLTR_BLANKINGFLTR7</code> | Blanking from another timing unit: TIMFLTR7 source  |
| <code>LL_HRTIM_EEFLTR_BLANKINGFLTR8</code> | Blanking from another timing unit: TIMFLTR8 source  |
| <code>LL_HRTIM_EEFLTR_WINDOWINGCMP2</code> | Windowing from counter reset/roll-over to Compare 2 |
| <code>LL_HRTIM_EEFLTR_WINDOWINGCMP3</code> | Windowing from counter reset/roll-over to Compare 3 |
| <code>LL_HRTIM_EEFLTR_WINDOWINGTIM</code>  | Windowing from another timing unit: TIMWIN source   |

**TIMER EXTERNAL EVENT LATCH STATUS**

|                                        |                                                                                  |
|----------------------------------------|----------------------------------------------------------------------------------|
| <code>LL_HRTIM_EELATCH_DISABLED</code> | Event is ignored if it happens during a blank, or passed through during a window |
| <code>LL_HRTIM_EELATCH_ENABLED</code>  | Event is latched and delayed till the end of the blanking or windowing period    |

**UPDATE GATING**

|                                                    |                                                                                |
|----------------------------------------------------|--------------------------------------------------------------------------------|
| <code>LL_HRTIM_UPDATEGATING_INDEPENDENT</code>     | Update done independently from the DMA burst transfer completion               |
| <code>LL_HRTIM_UPDATEGATING_DMABURST</code>        | Update done when the DMA burst transfer is completed                           |
| <code>LL_HRTIM_UPDATEGATING_DMABURST_UPDATE</code> | Update done on timer roll-over following a DMA burst transfer completion       |
| <code>LL_HRTIM_UPDATEGATING_UPDEN1</code>          | Slave timer only - Update done on a rising edge of HRTIM update enable input 1 |
| <code>LL_HRTIM_UPDATEGATING_UPDEN2</code>          | Slave timer only - Update done on a rising edge of HRTIM update                |

|                                     |                                                                                                           |
|-------------------------------------|-----------------------------------------------------------------------------------------------------------|
|                                     | enable input 2                                                                                            |
| LL_HRTIM_UPDATEGATING_UPDEN3        | Slave timer only - Update done on a rising edge of HRTIM update enable input 3                            |
| LL_HRTIM_UPDATEGATING_UPDEN1_UPDATE | Slave timer only - Update done on the update event following a rising edge of HRTIM update enable input 1 |
| LL_HRTIM_UPDATEGATING_UPDEN2_UPDATE | Slave timer only - Update done on the update event following a rising edge of HRTIM update enable input 2 |
| LL_HRTIM_UPDATEGATING_UPDEN3_UPDATE | Slave timer only - Update done on the update event following a rising edge of HRTIM update enable input 3 |

***UPDATE TRIGGER***

|                                |                                                                                                                    |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------|
| LL_HRTIM_UPDATETRIG_NONE       | Register update is disabled                                                                                        |
| LL_HRTIM_UPDATETRIG_MASTER     | Register update is triggered by the master timer update                                                            |
| LL_HRTIM_UPDATETRIG_TIMER_A    | Register update is triggered by the timer A update                                                                 |
| LL_HRTIM_UPDATETRIG_TIMER_B    | Register update is triggered by the timer B update                                                                 |
| LL_HRTIM_UPDATETRIG_TIMER_C    | Register update is triggered by the timer C update                                                                 |
| LL_HRTIM_UPDATETRIG_TIMER_D    | Register update is triggered by the timer D update                                                                 |
| LL_HRTIM_UPDATETRIG_TIMER_E    | Register update is triggered by the timer E update                                                                 |
| LL_HRTIM_UPDATETRIG_REPETITION | Register update is triggered when the counter rolls over and HRTIM_REPx = 0                                        |
| LL_HRTIM_UPDATETRIG_RESET      | Register update is triggered by counter reset or roll-over to 0 after reaching the period value in continuous mode |

***Exported Macros***

| <u>__LL_HRTIM_GET_OUTPUT__</u>                                                                                                                         | <b>Description:</b>                                                                                                            |
|--------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| STATE                                                                                                                                                  | <ul style="list-style-type: none"> <li>• HELPER macro returning the output state from output enable/disable status.</li> </ul> |
| <b>Parameters:</b>                                                                                                                                     |                                                                                                                                |
| <ul style="list-style-type: none"> <li>• __OUTPUT_STATUS_EN__: output enable status</li> <li>• __OUTPUT_STATUS_DIS__: output Disable status</li> </ul> |                                                                                                                                |
| <b>Return value:</b>                                                                                                                                   |                                                                                                                                |
| <ul style="list-style-type: none"> <li>• Returned: value can be one of the following values:</li> </ul>                                                |                                                                                                                                |

- LL\_HRTIM\_OUTPUTSTATE\_IDLE
- LL\_HRTIM\_OUTPUTSTATE\_RUN
- LL\_HRTIM\_OUTPUTSTATE\_FAULT

**Common Write and read registers Macros****LL\_HRTIM\_WriteReg      Description:**

- Write a value in HRTIM register.

**Parameters:**

- \_\_INSTANCE\_\_: HRTIM Instance
- \_\_REG\_\_: Register to be written
- \_\_VALUE\_\_: Value to be written in the register

**Return value:**

- None

**LL\_HRTIM\_ReadReg****Description:**

- Read a value in HRTIM register.

**Parameters:**

- \_\_INSTANCE\_\_: HRTIM Instance
- \_\_REG\_\_: Register to be read

**Return value:**

- Register: value

## 69 LL I2C Generic Driver

### 69.1 I2C Firmware driver registers structures

#### 69.1.1 LL\_I2C\_InitTypeDef

##### Data Fields

- *uint32\_t PeripheralMode*
- *uint32\_t Timing*
- *uint32\_t AnalogFilter*
- *uint32\_t DigitalFilter*
- *uint32\_t OwnAddress1*
- *uint32\_t TypeAcknowledge*
- *uint32\_t OwnAddrSize*

##### Field Documentation

- ***uint32\_t LL\_I2C\_InitTypeDef::PeripheralMode***  
Specifies the peripheral mode. This parameter can be a value of **I2C\_LL\_EC\_PERIPHERAL\_MODE**This feature can be modified afterwards using unitary function **LL\_I2C\_SetMode()**.
- ***uint32\_t LL\_I2C\_InitTypeDef::Timing***  
Specifies the SDA setup, hold time and the SCL high, low period values. This parameter must be set by referring to the STM32CubeMX Tool and the helper macro **\_LL\_I2C\_CONVERT\_TIMINGS()**This feature can be modified afterwards using unitary function **LL\_I2C\_SetTiming()**.
- ***uint32\_t LL\_I2C\_InitTypeDef::AnalogFilter***  
Enables or disables analog noise filter. This parameter can be a value of **I2C\_LL\_EC\_ANALOGFILTER\_SELECTION**This feature can be modified afterwards using unitary functions **LL\_I2C\_EnableAnalogFilter()** or **LL\_I2C\_DisableAnalogFilter()**.
- ***uint32\_t LL\_I2C\_InitTypeDef::DigitalFilter***  
Configures the digital noise filter. This parameter can be a number between Min\_Data = 0x00 and Max\_Data = 0x0FThis feature can be modified afterwards using unitary function **LL\_I2C\_SetDigitalFilter()**.
- ***uint32\_t LL\_I2C\_InitTypeDef::OwnAddress1***  
Specifies the device own address 1. This parameter must be a value between Min\_Data = 0x00 and Max\_Data = 0x3FFThis feature can be modified afterwards using unitary function **LL\_I2C\_SetOwnAddress1()**.
- ***uint32\_t LL\_I2C\_InitTypeDef::TypeAcknowledge***  
Specifies the ACKnowledge or Non ACKnowledge condition after the address receive match code or next received byte. This parameter can be a value of **I2C\_LL\_EC\_I2C\_ACKNOWLEDGE**This feature can be modified afterwards using unitary function **LL\_I2C\_AcknowledgeNextData()**.
- ***uint32\_t LL\_I2C\_InitTypeDef::OwnAddrSize***  
Specifies the device own address 1 size (7-bit or 10-bit). This parameter can be a value of **I2C\_LL\_EC\_OWNADDRESS1**This feature can be modified afterwards using unitary function **LL\_I2C\_SetOwnAddress1()**.

## 69.2 I2C Firmware driver API description

### 69.2.1 Detailed description of functions

#### LL\_I2C\_Enable

|                                                   |                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_Enable (I2C_TypeDef * I2Cx)</code>           |
| Function description                              | Enable I2C peripheral (PE = 1).                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 PE LL_I2C_Enable</li> </ul>       |

#### LL\_I2C\_Disable

|                                                   |                                                                                                                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_Disable (I2C_TypeDef * I2Cx)</code>                                                                                                                                                                                  |
| Function description                              | Disable I2C peripheral (PE = 0).                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• When PE = 0, the I2C SCL and SDA lines are released. Internal state machines and status bits are put back to their reset value. When cleared, PE must be kept low for at least 3 APB clock cycles.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 PE LL_I2C_Disable</li> </ul>                                                                                                                                                                              |

#### LL\_I2C\_IsEnabled

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabled (I2C_TypeDef * I2Cx)</code>        |
| Function description                              | Check if the I2C peripheral is enabled or disabled.                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 PE LL_I2C_IsEnabled</li> </ul>        |

#### LL\_I2C\_ConfigFilters

|                      |                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_ConfigFilters (I2C_TypeDef * I2Cx, uint32_t AnalogFilter, uint32_t DigitalFilter)</code>                                |
| Function description | Configure Noise Filters (Analog and Digital).                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>AnalogFilter:</b> This parameter can be one of the following</li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <p>values:</p> <ul style="list-style-type: none"> <li>- LL_I2C_ANALOGFILTER_ENABLE</li> <li>- LL_I2C_ANALOGFILTER_DISABLE</li> </ul>                                                                                                                                                                                                                                                                                                   |
|                                                   | <ul style="list-style-type: none"> <li>• <b>DigitalFilter:</b> This parameter must be a value between Min_Data=0x00 (Digital filter disabled) and Max_Data=0x0F (Digital filter enabled and filtering capability up to 15*t<sub>i2cclk</sub>). This parameter is used to configure the digital noise filter on SDA and SCL input. The digital filter will filter spikes with a length of up to DNF[3:0]*t<sub>i2cclk</sub>.</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• If the analog filter is also enabled, the digital filter is added to analog filter. The filters can only be programmed when the I2C is disabled (PE = 0).</li> </ul>                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 ANFOFF LL_I2C_ConfigFilters</li> <li>• CR1 DNF LL_I2C_ConfigFilters</li> </ul>                                                                                                                                                                                                                                                                                                            |

### LL\_I2C\_SetDigitalFilter

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_SetDigitalFilter (I2C_TypeDef *<br/>I2Cx, uint32_t DigitalFilter)</code>                                                                                                                                                                                                                                                                                                                                                                   |
| Function description                              | Configure Digital Noise Filter.                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>DigitalFilter:</b> This parameter must be a value between Min_Data=0x00 (Digital filter disabled) and Max_Data=0x0F (Digital filter enabled and filtering capability up to 15*t<sub>i2cclk</sub>). This parameter is used to configure the digital noise filter on SDA and SCL input. The digital filter will filter spikes with a length of up to DNF[3:0]*t<sub>i2cclk</sub>.</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• If the analog filter is also enabled, the digital filter is added to analog filter. This filter can only be programmed when the I2C is disabled (PE = 0).</li> </ul>                                                                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 DNF LL_I2C_SetDigitalFilter</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                          |

### LL\_I2C\_GetDigitalFilter

|                                                   |                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_GetDigitalFilter<br/>(I2C_TypeDef * I2Cx)</code>                  |
| Function description                              | Get the current Digital Noise Filter configuration.                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x0 and Max_Data=0xF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 DNF LL_I2C_GetDigitalFilter</li> </ul>                     |

**LL\_I2C\_EnableAnalogFilter**

|                                                   |                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_EnableAnalogFilter(<br/>I2C_TypeDef * I2Cx)</code>                                      |
| Function description                              | Enable Analog Noise Filter.                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• This filter can only be programmed when the I2C is disabled (PE = 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 ANFOFF LL_I2C_EnableAnalogFilter</li> </ul>                                  |

**LL\_I2C\_DisableAnalogFilter**

|                                                   |                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_DisableAnalogFilter(<br/>I2C_TypeDef * I2Cx)</code>                                     |
| Function description                              | Disable Analog Noise Filter.                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• This filter can only be programmed when the I2C is disabled (PE = 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 ANFOFF LL_I2C_DisableAnalogFilter</li> </ul>                                 |

**LL\_I2C\_IsEnabledAnalogFilter**

|                                                   |                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledAnalogFilter(<br/>I2C_TypeDef * I2Cx)</code> |
| Function description                              | Check if Analog Noise Filter is enabled or disabled.                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 ANFOFF LL_I2C_IsEnabledAnalogFilter</li> </ul> |

**LL\_I2C\_EnableDMAReq\_TX**

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_EnableDMAReq_TX(<br/>I2C_TypeDef * I2Cx)</code> |
| Function description | Enable DMA transmission requests.                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                   |

- Reference Manual to  
LL API cross  
reference:
- CR1 TXDMAEN LL\_I2C\_EnableDMAReq\_TX

### LL\_I2C\_DisableDMAReq\_TX

Function name `__STATIC_INLINE void LL_I2C_DisableDMAReq_TX(I2C_TypeDef * I2Cx)`

Function description Disable DMA transmission requests.

- Parameters
- **I2Cx:** I2C Instance.

- Return values
- **None**

- Reference Manual to  
LL API cross  
reference:
- CR1 TXDMAEN LL\_I2C\_DisableDMAReq\_TX

### LL\_I2C\_IsEnabledDMAReq\_TX

Function name `__STATIC_INLINE uint32_t LL_I2C_IsEnabledDMAReq_TX(I2C_TypeDef * I2Cx)`

Function description Check if DMA transmission requests are enabled or disabled.

- Parameters
- **I2Cx:** I2C Instance.

- Return values
- **State:** of bit (1 or 0).

- Reference Manual to  
LL API cross  
reference:
- CR1 TXDMAEN LL\_I2C\_IsEnabledDMAReq\_TX

### LL\_I2C\_EnableDMAReq\_RX

Function name `__STATIC_INLINE void LL_I2C_EnableDMAReq_RX(I2C_TypeDef * I2Cx)`

Function description Enable DMA reception requests.

- Parameters
- **I2Cx:** I2C Instance.

- Return values
- **None**

- Reference Manual to  
LL API cross  
reference:
- CR1 RXDMAEN LL\_I2C\_EnableDMAReq\_RX

### LL\_I2C\_DisableDMAReq\_RX

Function name `__STATIC_INLINE void LL_I2C_DisableDMAReq_RX(I2C_TypeDef * I2Cx)`

Function description Disable DMA reception requests.

- Parameters
- **I2Cx:** I2C Instance.

- Return values
- **None**

- Reference Manual to  
LL API cross  
reference:
- CR1 RXDMAEN LL\_I2C\_DisableDMAReq\_RX

### **LL\_I2C\_IsEnabledDMAReq\_RX**

|                      |                                                                                          |
|----------------------|------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledDMAReq_RX(<br/>I2C_TypeDef * I2Cx)</code> |
| Function description | Check if DMA reception requests are enabled or disabled.                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>       |

- Reference Manual to  
LL API cross  
reference:
- CR1 RXDMAEN LL\_I2C\_IsEnabledDMAReq\_RX

### **LL\_I2C\_DMA\_GetRegAddr**

|                                                   |                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_DMA_GetRegAddr(<br/>I2C_TypeDef * I2Cx, uint32_t Direction)</code>                                                                                                                                                                               |
| Function description                              | Get the data register address used for DMA transfer.                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance</li> <li>• <b>Direction:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_I2C_DMA_REG_DATA_TRANSMIT</li> <li>– LL_I2C_DMA_REG_DATA_RECEIVE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Address:</b> of data register</li> </ul>                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TXDR TXDATA LL_I2C_DMA_GetRegAddr</li> <li>• RXDR RXDATA LL_I2C_DMA_GetRegAddr</li> </ul>                                                                                                                                                     |

### **LL\_I2C\_EnableClockStretching**

|                                                   |                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_EnableClockStretching(<br/>I2C_TypeDef * I2Cx)</code>                                |
| Function description                              | Enable Clock stretching.                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit can only be programmed when the I2C is disabled (PE = 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 NOSTRETCH LL_I2C_EnableClockStretching</li> </ul>                         |

**LL\_I2C\_DisableClockStretching**

|                                             |                                                                                                                    |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE void LL_I2C_DisableClockStretching(I2C_TypeDef * I2Cx)</code></b>                         |
| Function description                        | Disable Clock stretching.                                                                                          |
| Parameters                                  | <ul style="list-style-type: none"><li><b>I2Cx:</b> I2C Instance.</li></ul>                                         |
| Return values                               | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                        |
| Notes                                       | <ul style="list-style-type: none"><li>This bit can only be programmed when the I2C is disabled (PE = 0).</li></ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>CR1 NOSTRETCH LL_I2C_DisableClockStretching</li></ul>                        |

**LL\_I2C\_IsEnabledClockStretching**

|                                             |                                                                                                  |
|---------------------------------------------|--------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledClockStretching(I2C_TypeDef * I2Cx)</code></b> |
| Function description                        | Check if Clock stretching is enabled or disabled.                                                |
| Parameters                                  | <ul style="list-style-type: none"><li><b>I2Cx:</b> I2C Instance.</li></ul>                       |
| Return values                               | <ul style="list-style-type: none"><li><b>State:</b> of bit (1 or 0).</li></ul>                   |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>CR1 NOSTRETCH LL_I2C_IsEnabledClockStretching</li></ul>    |

**LL\_I2C\_EnableSlaveByteControl**

|                                             |                                                                                            |
|---------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE void LL_I2C_EnableSlaveByteControl(I2C_TypeDef * I2Cx)</code></b> |
| Function description                        | Enable hardware byte control in slave mode.                                                |
| Parameters                                  | <ul style="list-style-type: none"><li><b>I2Cx:</b> I2C Instance.</li></ul>                 |
| Return values                               | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>CR1 SBC LL_I2C_EnableSlaveByteControl</li></ul>      |

**LL\_I2C\_DisableSlaveByteControl**

|                                             |                                                                                             |
|---------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE void LL_I2C_DisableSlaveByteControl(I2C_TypeDef * I2Cx)</code></b> |
| Function description                        | Disable hardware byte control in slave mode.                                                |
| Parameters                                  | <ul style="list-style-type: none"><li><b>I2Cx:</b> I2C Instance.</li></ul>                  |
| Return values                               | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                 |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>CR1 SBC LL_I2C_DisableSlaveByteControl</li></ul>      |

**LL\_I2C\_IsEnabledSlaveByteControl**

|                                                   |                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_I2C_IsEnabledSlaveByteControl (I2C_TypeDef * I2Cx)</code> |
| Function description                              | Check if hardware byte control in slave mode is enabled or disabled.                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 SBC LL_I2C_IsEnabledSlaveByteControl</li> </ul>    |

**LL\_I2C\_EnableWakeUpFromStop**

|                                                   |                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_EnableWakeUpFromStop<br/>(I2C_TypeDef * I2Cx)</code>                                                                                                                                                                                    |
| Function description                              | Enable Wakeup from STOP.                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_I2C_WAKEUP_FROMSTOP_INSTANCE(I2Cx) can be used to check whether or not WakeUpFromStop feature is supported by the I2Cx Instance.</li> <li>• This bit can only be programmed when Digital Filter is disabled.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 WUPEN LL_I2C_EnableWakeUpFromStop</li> </ul>                                                                                                                                                                                 |

**LL\_I2C\_DisableWakeUpFromStop**

|                                                   |                                                                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_DisableWakeUpFromStop<br/>(I2C_TypeDef * I2Cx)</code>                                                                                                       |
| Function description                              | Disable Wakeup from STOP.                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_I2C_WAKEUP_FROMSTOP_INSTANCE(I2Cx) can be used to check whether or not WakeUpFromStop feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 WUPEN LL_I2C_DisableWakeUpFromStop</li> </ul>                                                                                                    |

**LL\_I2C\_IsEnabledWakeUpFromStop**

|                                             |                                                                                                                                                                                               |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledWakeUpFromStop (I2C_TypeDef * I2Cx)</code>                                                                                                     |
| Function description                        | Check if Wakeup from STOP is enabled or disabled.                                                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                            |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro IS_I2C_WAKEUP_FROMSTOP_INSTANCE(I2Cx) can be used to check whether or not WakeUpFromStop feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR1 WUPEN LL_I2C_IsEnabledWakeUpFromStop</li> </ul>                                                                                                  |

**LL\_I2C\_EnableGeneralCall**

|                                             |                                                                                             |
|---------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_I2C_EnableGeneralCall (I2C_TypeDef * I2Cx)</code>             |
| Function description                        | Enable General Call.                                                                        |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>              |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                             |
| Notes                                       | <ul style="list-style-type: none"> <li>• When enabled the Address 0x00 is ACKed.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR1 GCEN LL_I2C_EnableGeneralCall</li> </ul>       |

**LL\_I2C\_DisableGeneralCall**

|                                             |                                                                                               |
|---------------------------------------------|-----------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_I2C_DisableGeneralCall (I2C_TypeDef * I2Cx)</code>              |
| Function description                        | Disable General Call.                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                               |
| Notes                                       | <ul style="list-style-type: none"> <li>• When disabled the Address 0x00 is NACKed.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR1 GCEN LL_I2C_DisableGeneralCall</li> </ul>        |

**LL\_I2C\_IsEnabledGeneralCall**

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledGeneralCall (I2C_TypeDef * I2Cx)</code> |
| Function description | Check if General Call is enabled or disabled.                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>         |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>     |

- Reference Manual to  
LL API cross  
reference:
- CR1 GCEN LL\_I2C\_IsEnabledGeneralCall

### LL\_I2C\_SetMasterAddressingMode

|                                                   |                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_SetMasterAddressingMode(I2C_TypeDef * I2Cx, uint32_t AddressingMode)</code>                                                                                                                                                                                          |
| Function description                              | Configure the Master to operate in 7-bit or 10-bit addressing mode.                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>AddressingMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_I2C_ADDRESSING_MODE_7BIT</li> <li>– LL_I2C_ADDRESSING_MODE_10BIT</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• Changing this bit is not allowed, when the START bit is set.</li> </ul>                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 ADD10 LL_I2C_SetMasterAddressingMode</li> </ul>                                                                                                                                                                                                           |

### LL\_I2C\_GetMasterAddressingMode

|                                                   |                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_GetMasterAddressingMode(I2C_TypeDef * I2Cx)</code>                                                                                                                                                          |
| Function description                              | Get the Master addressing mode.                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_I2C_ADDRESSING_MODE_7BIT</li> <li>– LL_I2C_ADDRESSING_MODE_10BIT</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 ADD10 LL_I2C_GetMasterAddressingMode</li> </ul>                                                                                                                                                      |

### LL\_I2C\_SetOwnAddress1

|                      |                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_SetOwnAddress1(I2C_TypeDef * I2Cx, uint32_t OwnAddress1, uint32_t OwnAddrSize)</code>                                                                                                                                                                                                                                                                          |
| Function description | Set the Own Address1.                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>OwnAddress1:</b> This parameter must be a value between Min_Data=0 and Max_Data=0x3FF.</li> <li>• <b>OwnAddrSize:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_I2C_OWNADDRESS1_7BIT</li> <li>– LL_I2C_OWNADDRESS1_10BIT</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                  |

- Reference Manual to  
LL API cross  
reference:
- OAR1 OA1 LL\_I2C\_SetOwnAddress1
  - OAR1 OA1MODE LL\_I2C\_SetOwnAddress1

### **LL\_I2C\_EnableOwnAddress1**

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_EnableOwnAddress1<br/>(I2C_TypeDef * I2Cx)</code>     |
| Function description                              | Enable acknowledge on Own Address1 match address.                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OAR1 OA1EN LL_I2C_EnableOwnAddress1</li> </ul> |

### **LL\_I2C\_DisableOwnAddress1**

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_DisableOwnAddress1<br/>(I2C_TypeDef * I2Cx)</code>     |
| Function description                              | Disable acknowledge on Own Address1 match address.                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OAR1 OA1EN LL_I2C_DisableOwnAddress1</li> </ul> |

### **LL\_I2C\_IsEnabledOwnAddress1**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledOwnAddress1<br/>(I2C_TypeDef * I2Cx)</code> |
| Function description                              | Check if Own Address1 acknowledge is enabled or disabled.                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• OAR1 OA1EN LL_I2C_IsEnabledOwnAddress1</li> </ul> |

### **LL\_I2C\_SetOwnAddress2**

|                      |                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_SetOwnAddress2<br/>(I2C_TypeDef * I2Cx, uint32_t OwnAddress2, uint32_t<br/>OwnAddrMask)</code>                          |
| Function description | Set the 7bits Own Address2.                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>OwnAddress2:</b> Value between Min_Data=0 and Max_Data=0x7F.</li> </ul> |

- **OwnAddrMask:** This parameter can be one of the following values:
  - LL\_I2C\_OWNADDRESS2\_NOMASK
  - LL\_I2C\_OWNADDRESS2\_MASK01
  - LL\_I2C\_OWNADDRESS2\_MASK02
  - LL\_I2C\_OWNADDRESS2\_MASK03
  - LL\_I2C\_OWNADDRESS2\_MASK04
  - LL\_I2C\_OWNADDRESS2\_MASK05
  - LL\_I2C\_OWNADDRESS2\_MASK06
  - LL\_I2C\_OWNADDRESS2\_MASK07

Return values

- **None**

Notes

- This action has no effect if own address2 is enabled.

Reference Manual to  
LL API cross  
reference:

- OAR2 OA2 LL\_I2C\_SetOwnAddress2
- OAR2 OA2MSK LL\_I2C\_SetOwnAddress2

### **LL\_I2C\_EnableOwnAddress2**

Function name      **STATIC\_INLINE void LL\_I2C\_EnableOwnAddress2(I2C\_TypeDef \* I2Cx)**

Function description      Enable acknowledge on Own Address2 match address.

Parameters      • **I2Cx:** I2C Instance.

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- OAR2 OA2EN LL\_I2C\_EnableOwnAddress2

### **LL\_I2C\_DisableOwnAddress2**

Function name      **STATIC\_INLINE void LL\_I2C\_DisableOwnAddress2(I2C\_TypeDef \* I2Cx)**

Function description      Disable acknowledge on Own Address2 match address.

Parameters      • **I2Cx:** I2C Instance.

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- OAR2 OA2EN LL\_I2C\_DisableOwnAddress2

### **LL\_I2C\_IsEnabledOwnAddress2**

Function name      **STATIC\_INLINE uint32\_t LL\_I2C\_IsEnabledOwnAddress2(I2C\_TypeDef \* I2Cx)**

Function description      Check if Own Address1 acknowledge is enabled or disabled.

Parameters      • **I2Cx:** I2C Instance.

Return values

- **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- OAR2 OA2EN LL\_I2C\_IsEnabledOwnAddress2

### LL\_I2C\_SetTiming

Function name

**\_STATIC\_INLINE void LL\_I2C\_SetTiming (I2C\_TypeDef \*  
I2Cx, uint32\_t Timing)**

Function description

Configure the SDA setup, hold time and the SCL high, low period.

Parameters

- **I2Cx:** I2C Instance.
- **Timing:** This parameter must be a value between Min\_Data=0 and Max\_Data=0xFFFFFFFF.

Return values

- **None**

Notes

- This bit can only be programmed when the I2C is disabled (PE = 0).
- This parameter is computed with the STM32CubeMX Tool.

Reference Manual to  
LL API cross  
reference:

- TIMINGR TIMINGR LL\_I2C\_SetTiming

### LL\_I2C\_GetTimingPrescaler

Function name

**\_STATIC\_INLINE uint32\_t LL\_I2C\_GetTimingPrescaler  
(I2C\_TypeDef \* I2Cx)**

Function description

Get the Timing Prescaler setting.

Parameters

- **I2Cx:** I2C Instance.

Return values

- **Value:** between Min\_Data=0x0 and Max\_Data=0xF

Reference Manual to  
LL API cross  
reference:

- TIMINGR PRESC LL\_I2C\_GetTimingPrescaler

### LL\_I2C\_GetClockLowPeriod

Function name

**\_STATIC\_INLINE uint32\_t LL\_I2C\_GetClockLowPeriod  
(I2C\_TypeDef \* I2Cx)**

Function description

Get the SCL low period setting.

Parameters

- **I2Cx:** I2C Instance.

Return values

- **Value:** between Min\_Data=0x00 and Max\_Data=0xFF

Reference Manual to  
LL API cross  
reference:

- TIMINGR SCLL LL\_I2C\_GetClockLowPeriod

**LL\_I2C\_GetClockHighPeriod**

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_I2C_GetClockHighPeriod<br/>(I2C_TypeDef * I2Cx)</code></b>            |
| Function description                              | Get the SCL high period setting.                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMINGR SCLH LL_I2C_GetClockHighPeriod</li> </ul>                |

**LL\_I2C\_GetDataHoldTime**

|                                                   |                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_I2C_GetDataHoldTime<br/>(I2C_TypeDef * I2Cx)</code></b>             |
| Function description                              | Get the SDA hold time.                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x0 and Max_Data=0xF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMINGR SDADEL LL_I2C_GetDataHoldTime</li> </ul>               |

**LL\_I2C\_GetDataSetupTime**

|                                                   |                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_I2C_GetDataSetupTime<br/>(I2C_TypeDef * I2Cx)</code></b>            |
| Function description                              | Get the SDA setup time.                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x0 and Max_Data=0xF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMINGR SCLDEL LL_I2C_GetDataSetupTime</li> </ul>              |

**LL\_I2C\_SetMode**

|                      |                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_I2C_SetMode (I2C_TypeDef * I2Cx,<br/>uint32_t PeripheralMode)</code></b>                                                                                                                                                                                                                                               |
| Function description | Configure peripheral mode.                                                                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>PeripheralMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_I2C_MODE_I2C</li> <li>- LL_I2C_MODE_SMBUS_HOST</li> <li>- LL_I2C_MODE_SMBUS_DEVICE</li> <li>- LL_I2C_MODE_SMBUS_DEVICE_ARP</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                        |

|                                             |                                                                                                                                                                          |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 SMBHEN LL_I2C_SetMode</li> <li>CR1 SMBDEN LL_I2C_SetMode</li> </ul>                                                           |

### LL\_I2C\_GetMode

|                                             |                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_I2C_GetMode (I2C_TypeDef * I2Cx)</code>                                                                                                                                                                                                                                 |
| Function description                        | Get peripheral mode.                                                                                                                                                                                                                                                                                      |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                                                                              |
| Return values                               | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_I2C_MODE_I2C</li> <li>- LL_I2C_MODE_SMBUS_HOST</li> <li>- LL_I2C_MODE_SMBUS_DEVICE</li> <li>- LL_I2C_MODE_SMBUS_DEVICE_ARP</li> </ul> </li> </ul> |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul>                                                                                                                                  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 SMBHEN LL_I2C_GetMode</li> <li>CR1 SMBDEN LL_I2C_GetMode</li> </ul>                                                                                                                                                                                            |

### LL\_I2C\_EnableSMBusAlert

|                                             |                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_I2C_EnableSMBusAlert (I2C_TypeDef * I2Cx)</code>                                                                                                                                                                                                                                                                         |
| Function description                        | Enable SMBus alert (Host or Device mode)                                                                                                                                                                                                                                                                                                               |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                                                                                                                           |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                          |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> <li>SMBus Device mode: SMBus Alert pin is driven low and Alert Response Address Header acknowledge is enabled. SMBus Host mode: SMBus Alert pin management is supported.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 ALERTEN LL_I2C_EnableSMBusAlert</li> </ul>                                                                                                                                                                                                                                                                  |

### LL\_I2C\_DisableSMBusAlert

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_DisableSMBusAlert (I2C_TypeDef * I2Cx)</code> |
| Function description | Disable SMBus alert (Host or Device mode)                                       |

---

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                                                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> <li>SMBus Device mode: SMBus Alert pin is not driven (can be used as a standard GPIO) and Alert Response Address Header acknowledge is disabled. SMBus Host mode:SMBus Alert pin management is not supported.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 ALERTEN LL_I2C_DisableSMBusAlert</li> </ul>                                                                                                                                                                                                                                                                                                      |

### LL\_I2C\_IsEnabledSMBusAlert

|                                                   |                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledSMBusAlert(<br/>    I2C_TypeDef * I2Cx)</code>                                                                            |
| Function description                              | Check if SMBus alert (Host or Device mode) is enabled or disabled.                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 ALERTEN LL_I2C_IsEnabledSMBusAlert</li> </ul>                                                                                 |

### LL\_I2C\_EnableSMBusPEC

|                                                   |                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_EnableSMBusPEC(<br/>    I2C_TypeDef * I2Cx)</code>                                                                                     |
| Function description                              | Enable SMBus Packet Error Calculation (PEC).                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 PECEN LL_I2C_EnableSMBusPEC</li> </ul>                                                                                        |

### LL\_I2C\_DisableSMBusPEC

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_DisableSMBusPEC(<br/>    I2C_TypeDef * I2Cx)</code> |
| Function description | Disable SMBus Packet Error Calculation (PEC).                                         |

|                                             |                                                                                                                                                                          |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                  | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                             |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                            |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 PECEN LL_I2C_DisableSMBusPEC</li> </ul>                                                                                       |

### LL\_I2C\_IsEnabledSMBusPEC

|                                             |                                                                                                                                                                          |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE uint32_t LL_I2C_IsEnabledSMBusPEC(I2C_TypeDef * I2Cx)</code>                                                                                        |
| Function description                        | Check if SMBus Packet Error Calculation (PEC) is enabled or disabled.                                                                                                    |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                             |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                         |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 PECEN LL_I2C_IsEnabledSMBusPEC</li> </ul>                                                                                     |

### LL\_I2C\_ConfigSMBusTimeout

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE void LL_I2C_ConfigSMBusTimeout(I2C_TypeDef * I2Cx, uint32_t TimeoutA, uint32_t TimeoutAMode, uint32_t TimeoutB)</code>                                                                                                                                                                                                                                                                                     |
| Function description                        | Configure the SMBus Clock Timeout.                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> <li><b>TimeoutA:</b> This parameter must be a value between Min_Data=0 and Max_Data=0xFFFF.</li> <li><b>TimeoutAMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_I2C_SMBUS_TIMEOUTA_MODE_SCL_LOW</li> <li>LL_I2C_SMBUS_TIMEOUTA_MODE_SDA_SCL_HIGH</li> </ul> </li> <li><b>TimeoutB:</b></li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                   |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> <li>This configuration can only be programmed when associated Timeout is disabled (TimeoutA and/orTimeoutB).</li> </ul>                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>TIMEOUTR TIMEOUTA LL_I2C_ConfigSMBusTimeout</li> <li>TIMEOUTR TIDLE LL_I2C_ConfigSMBusTimeout</li> <li>TIMEOUTR TIMEOUTB LL_I2C_ConfigSMBusTimeout</li> </ul>                                                                                                                                                                                                                            |

**LL\_I2C\_SetSMBusTimeoutA**

|                                                   |                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_SetSMBusTimeoutA(I2C_TypeDef * I2Cx, uint32_t TimeoutA)</code>                                                                                                                                                   |
| Function description                              | Configure the SMBus Clock TimeoutA (SCL low timeout or SCL and SDA high timeout depends on TimeoutA mode).                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>TimeoutA:</b> This parameter must be a value between Min_Data=0 and Max_Data=0xFFFF.</li> </ul>                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> <li>• These bits can only be programmed when TimeoutA is disabled.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMEOUTR TIMEOUTA LL_I2C_SetSMBusTimeoutA</li> </ul>                                                                                                                                                      |

**LL\_I2C\_GetSMBusTimeoutA**

|                                                   |                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_GetSMBusTimeoutA(I2C_TypeDef * I2Cx)</code>                                                                                          |
| Function description                              | Get the SMBus Clock TimeoutA setting.                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0 and Max_Data=0xFFFF</li> </ul>                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMEOUTR TIMEOUTA LL_I2C_GetSMBusTimeoutA</li> </ul>                                                                              |

**LL\_I2C\_SetSMBusTimeoutAMode**

|                      |                                                                                                                                                                                                                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_SetSMBusTimeoutAMode(I2C_TypeDef * I2Cx, uint32_t TimeoutAMode)</code>                                                                                                                                                                                                     |
| Function description | Set the SMBus Clock TimeoutA mode.                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>TimeoutAMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_I2C_SMBUS_TIMEOUTA_MODE_SCL_LOW</li> <li>- LL_I2C_SMBUS_TIMEOUTA_MODE_SDA_SCL_HIGH</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                              |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul>                                                                                                                                   |

Reference  
Manual to LL API  
cross reference:

- This bit can only be programmed when TimeoutA is disabled.
- TIMEOUTTR TIDLE LL\_I2C\_SetSMBusTimeoutAMode

### **LL\_I2C\_GetSMBusTimeoutAMode**

|                                                   |                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_GetSMBusTimeoutAMode(I2C_TypeDef * I2Cx)</code>                                                                                                                                                                               |
| Function description                              | Get the SMBus Clock TimeoutA mode.                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_I2C_SMBUS_TIMEOUTA_MODE_SCL_LOW</li> <li>- LL_I2C_SMBUS_TIMEOUTA_MODE_SDA_SCL_HIGH</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul>                                                                                          |
| Reference<br>Manual to LL API<br>cross reference: | <ul style="list-style-type: none"> <li>• TIMEOUTTR TIDLE LL_I2C_GetSMBusTimeoutAMode</li> </ul>                                                                                                                                                                     |

### **LL\_I2C\_SetSMBusTimeoutB**

|                                                   |                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_SetSMBusTimeoutB(I2C_TypeDef * I2Cx, uint32_t TimeoutB)</code>                                                                                                                                                   |
| Function description                              | Configure the SMBus Extended Cumulative Clock TimeoutB (Master or Slave mode).                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>TimeoutB:</b> This parameter must be a value between Min_Data=0 and Max_Data=0xFFFF.</li> </ul>                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> <li>• These bits can only be programmed when TimeoutB is disabled.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TIMEOUTTR TIMEOUTB LL_I2C_SetSMBusTimeoutB</li> </ul>                                                                                                                                                     |

### **LL\_I2C\_GetSMBusTimeoutB**

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_I2C_GetSMBusTimeoutB(I2C_TypeDef * I2Cx)</code> |
| Function description | Get the SMBus Extended Cumulative Clock TimeoutB setting.                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>    |

|                                             |                                                                                                                                                                          |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                               | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0 and Max_Data=0FFF</li> </ul>                                                                     |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>TIMEOUTUR TIMEOUTB LL_I2C_GetSMBusTimeoutB</li> </ul>                                                                             |

### LL\_I2C\_EnableSMBusTimeout

|                                             |                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_I2C_EnableSMBusTimeout(I2C_TypeDef * I2Cx, uint32_t ClockTimeout)</code>                                                                                                                                                                                                        |
| Function description                        | Enable the SMBus Clock Timeout.                                                                                                                                                                                                                                                                               |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> <li><b>ClockTimeout:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_I2C_SMBUS_TIMEOUTA</li> <li>– LL_I2C_SMBUS_TIMEOUTB</li> <li>– LL_I2C_SMBUS_ALL_TIMEOUT</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                 |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul>                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>TIMEOUTUR TIMOUTEN LL_I2C_EnableSMBusTimeout</li> <li>TIMEOUTUR TEXTEN LL_I2C_EnableSMBusTimeout</li> </ul>                                                                                                                                                            |

### LL\_I2C\_DisableSMBusTimeout

|                                             |                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_I2C_DisableSMBusTimeout(I2C_TypeDef * I2Cx, uint32_t ClockTimeout)</code>                                                                                                                                                                                                       |
| Function description                        | Disable the SMBus Clock Timeout.                                                                                                                                                                                                                                                                              |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> <li><b>ClockTimeout:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_I2C_SMBUS_TIMEOUTA</li> <li>– LL_I2C_SMBUS_TIMEOUTB</li> <li>– LL_I2C_SMBUS_ALL_TIMEOUT</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                 |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul>                                                                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>TIMEOUTUR TIMOUTEN LL_I2C_DisableSMBusTimeout</li> <li>TIMEOUTUR TEXTEN LL_I2C_DisableSMBusTimeout</li> </ul>                                                                                                                                                          |

**LL\_I2C\_IsEnabledSMBusTimeout**

|                                             |                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledSMBusTimeout(I2C_TypeDef * I2Cx, uint32_t ClockTimeout)</code>                                                                                                                                                                                                     |
| Function description                        | Check if the SMBus Clock Timeout is enabled or disabled.                                                                                                                                                                                                                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>ClockTimeout:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_I2C_SMBUS_TIMEOUTA</li> <li>– LL_I2C_SMBUS_TIMEOUTB</li> <li>– LL_I2C_SMBUS_ALL_TIMEOUT</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                                                |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul>                                                                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• TIMEOUTR TIMOUTEN LL_I2C_IsEnabledSMBusTimeout</li> <li>• TIMEOUTR TEXTEN LL_I2C_IsEnabledSMBusTimeout</li> </ul>                                                                                                                                                        |

**LL\_I2C\_EnableIT\_TX**

|                                             |                                                                                 |
|---------------------------------------------|---------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_I2C_EnableIT_TX(I2C_TypeDef * I2Cx)</code>        |
| Function description                        | Enable TXIS interrupt.                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>  |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                 |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR1 TXIE LL_I2C_EnableIT_TX</li> </ul> |

**LL\_I2C\_DisableIT\_TX**

|                                             |                                                                                  |
|---------------------------------------------|----------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_I2C_DisableIT_TX(I2C_TypeDef * I2Cx)</code>        |
| Function description                        | Disable TXIS interrupt.                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>   |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR1 TXIE LL_I2C_DisableIT_TX</li> </ul> |

**LL\_I2C\_IsEnabledIT\_TX**

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledIT_TX(I2C_TypeDef * I2Cx)</code> |
| Function description | Check if the TXIS Interrupt is enabled or disabled.                             |

---

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 TXIE LL_I2C_IsEnabledIT_TX</li> </ul> |

### LL\_I2C\_EnableIT\_RX

|                                                   |                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_EnableIT_RX (I2C_TypeDef * I2Cx)</code>     |
| Function description                              | Enable RXNE interrupt.                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>  |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 RXIE LL_I2C_EnableIT_RX</li> </ul> |

### LL\_I2C\_DisableIT\_RX

|                                                   |                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_DisableIT_RX (I2C_TypeDef * I2Cx)</code>     |
| Function description                              | Disable RXNE interrupt.                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>   |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 RXIE LL_I2C_DisableIT_RX</li> </ul> |

### LL\_I2C\_IsEnabledIT\_RX

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledIT_RX (I2C_TypeDef * I2Cx)</code> |
| Function description                              | Check if the RXNE Interrupt is enabled or disabled.                              |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 RXIE LL_I2C_IsEnabledIT_RX</li> </ul> |

### LL\_I2C\_EnableIT\_ADDR

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_EnableIT_ADDR (I2C_TypeDef * I2Cx)</code>  |
| Function description | Enable Address match interrupt (slave mode only).                            |
| Parameters           | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul> |

---

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 ADDRIE LL_I2C_EnableIT_ADDR</li> </ul> |

### LL\_I2C\_DisableIT\_ADDR

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_DisableIT_ADDR (I2C_TypeDef * I2Cx)</code>       |
| Function description                              | Disable Address match interrupt (slave mode only).                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 ADDRIE LL_I2C_DisableIT_ADDR</li> </ul> |

### LL\_I2C\_IsEnabledIT\_ADDR

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledIT_ADDR (I2C_TypeDef * I2Cx)</code>   |
| Function description                              | Check if Address match interrupt is enabled or disabled.                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 ADDRIE LL_I2C_IsEnabledIT_ADDR</li> </ul> |

### LL\_I2C\_EnableIT\_NACK

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_EnableIT_NACK (I2C_TypeDef * I2Cx)</code>       |
| Function description                              | Enable Not acknowledge received interrupt.                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 NACKIE LL_I2C_EnableIT_NACK</li> </ul> |

### LL\_I2C\_DisableIT\_NACK

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_DisableIT_NACK (I2C_TypeDef * I2Cx)</code> |
| Function description | Disable Not acknowledge received interrupt.                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                |

- Reference Manual to  
LL API cross  
reference:
- CR1 NACKIE LL\_I2C\_DisableIT\_NACK

### **LL\_I2C\_IsEnabledIT\_NACK**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_I2C_IsEnabledIT_NACK<br/>(I2C_TypeDef * I2Cx)</code>  |
| Function description                              | Check if Not acknowledge received interrupt is enabled or disabled.                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 NACKIE LL_I2C_IsEnabledIT_NACK</li> </ul> |

### **LL\_I2C\_EnableIT\_STOP**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_I2C_EnableIT_STOP (I2C_TypeDef<br/>* I2Cx)</code>      |
| Function description                              | Enable STOP detection interrupt.                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 STOPIE LL_I2C_EnableIT_STOP</li> </ul> |

### **LL\_I2C\_DisableIT\_STOP**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_I2C_DisableIT_STOP (I2C_TypeDef<br/>* I2Cx)</code>      |
| Function description                              | Disable STOP detection interrupt.                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 STOPIE LL_I2C_DisableIT_STOP</li> </ul> |

### **LL\_I2C\_IsEnabledIT\_STOP**

|                      |                                                                                       |
|----------------------|---------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_I2C_IsEnabledIT_STOP<br/>(I2C_TypeDef * I2Cx)</code> |
| Function description | Check if STOP detection interrupt is enabled or disabled.                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>    |

- Reference Manual to  
LL API cross  
reference:
- CR1 STOPIE LL\_I2C\_IsEnabledIT\_STOP

### **LL\_I2C\_EnableIT\_TC**

|                                                   |                                                                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_I2C_EnableIT_TC (I2C_TypeDef * I2Cx)</code></b>                                                                         |
| Function description                              | Enable Transfer Complete interrupt.                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• Any of these events will generate interrupt : Transfer Complete (TC) Transfer Complete Reload (TCR)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 TCIE LL_I2C_EnableIT_TC</li> </ul>                                                                         |

### **LL\_I2C\_DisableIT\_TC**

|                                                   |                                                                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_I2C_DisableIT_TC (I2C_TypeDef * I2Cx)</code></b>                                                                        |
| Function description                              | Disable Transfer Complete interrupt.                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• Any of these events will generate interrupt : Transfer Complete (TC) Transfer Complete Reload (TCR)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 TCIE LL_I2C_DisableIT_TC</li> </ul>                                                                        |

### **LL\_I2C\_IsEnabledIT\_TC**

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_I2C_IsEnabledIT_TC (I2C_TypeDef * I2Cx)</code></b> |
| Function description | Check if Transfer Complete interrupt is enabled or disabled.                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>         |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>     |

- Reference Manual to  
LL API cross  
reference:
- CR1 TCIE LL\_I2C\_IsEnabledIT\_TC

### **LL\_I2C\_EnableIT\_ERR**

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_I2C_EnableIT_ERR (I2C_TypeDef * I2Cx)</code></b> |
| Function description | Enable Error interrupts.                                                         |

---

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                                                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> <li>Any of these errors will generate interrupt : Arbitration Loss (ARLO) Bus Error detection (BERR) Overrun/Underrun (OVR) SMBus Timeout detection (TIMEOUT) SMBus PEC error detection (PECERR) SMBus Alert pin event detection (ALERT)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 ERRIE LL_I2C_EnableIT_ERR</li> </ul>                                                                                                                                                                                                                                                                                                                                        |

### LL\_I2C\_DisableIT\_ERR

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>STATIC_INLINE void LL_I2C_DisableIT_ERR (I2C_TypeDef * I2Cx)</code>                                                                                                                                                                                                                                                                                                                                              |
| Function description                              | Disable Error interrupts.                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                                                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> <li>Any of these errors will generate interrupt : Arbitration Loss (ARLO) Bus Error detection (BERR) Overrun/Underrun (OVR) SMBus Timeout detection (TIMEOUT) SMBus PEC error detection (PECERR) SMBus Alert pin event detection (ALERT)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 ERRIE LL_I2C_DisableIT_ERR</li> </ul>                                                                                                                                                                                                                                                                                                                                       |

### LL\_I2C\_IsEnabledIT\_ERR

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>STATIC_INLINE uint32_t LL_I2C_IsEnabledIT_ERR (I2C_TypeDef * I2Cx)</code>    |
| Function description                              | Check if Error interrupts are enabled or disabled.                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 ERRIE LL_I2C_IsEnabledIT_ERR</li> </ul> |

**LL\_I2C\_IsActiveFlag\_TXE**

|                                                   |                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_TXE<br/>(I2C_TypeDef * I2Cx)</code>                                                                              |
| Function description                              | Indicate the status of Transmit data register empty flag.                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• RESET: When next data is written in Transmit data register.</li> <li>SET: When Transmit data register is empty.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR TXE LL_I2C_IsActiveFlag_TXE</li> </ul>                                                                                 |

**LL\_I2C\_IsActiveFlag\_TXIS**

|                                                   |                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_TXIS<br/>(I2C_TypeDef * I2Cx)</code>                                                                             |
| Function description                              | Indicate the status of Transmit interrupt flag.                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• RESET: When next data is written in Transmit data register.</li> <li>SET: When Transmit data register is empty.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR TXIS LL_I2C_IsActiveFlag_TXIS</li> </ul>                                                                               |

**LL\_I2C\_IsActiveFlag\_RXNE**

|                                                   |                                                                                                                                                                |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_RXNE<br/>(I2C_TypeDef * I2Cx)</code>                                                                        |
| Function description                              | Indicate the status of Receive data register not empty flag.                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• RESET: When Receive data register is read. SET: When the received data is copied in Receive data register.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR RXNE LL_I2C_IsActiveFlag_RXNE</li> </ul>                                                                          |

**LL\_I2C\_IsActiveFlag\_ADDR**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_ADDR<br/>(I2C_TypeDef * I2Cx)</code> |
| Function description | Indicate the status of Address matched flag (slave mode).                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>          |

---

|                                                   |                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>RESET: Clear default value. SET: When the received slave address matched with one of the enabled slave address.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR ADDR LL_I2C_IsActiveFlag_ADDR</li> </ul>                                                                               |

### LL\_I2C\_IsActiveFlag\_NACK

|                                                   |                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_NACK<br/>(I2C_TypeDef * I2Cx)</code>                                                |
| Function description                              | Indicate the status of Not Acknowledge received flag.                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>RESET: Clear default value. SET: When a NACK is received after a byte transmission.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR NACKF LL_I2C_IsActiveFlag_NACK</li> </ul>                                                  |

### LL\_I2C\_IsActiveFlag\_STOP

|                                                   |                                                                                                                       |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_STOP<br/>(I2C_TypeDef * I2Cx)</code>                                |
| Function description                              | Indicate the status of Stop detection flag.                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                          |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>RESET: Clear default value. SET: When a Stop condition is detected.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR STOPF LL_I2C_IsActiveFlag_STOP</li> </ul>                                  |

### LL\_I2C\_IsActiveFlag\_TC

|                                                   |                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_TC<br/>(I2C_TypeDef * I2Cx)</code>                                                               |
| Function description                              | Indicate the status of Transfer complete flag (master mode).                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>RESET: Clear default value. SET: When RELOAD=0, AUTOEND=0 and NBYTES date have been transferred.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR TC LL_I2C_IsActiveFlag_TC</li> </ul>                                                                    |

**LL\_I2C\_IsActiveFlag\_TCR**

|                                                   |                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_TCR<br/>(I2C_TypeDef * I2Cx)</code>                                                    |
| Function description                              | Indicate the status of Transfer complete flag (master mode).                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• RESET: Clear default value. SET: When RELOAD=1 and NBYTES date have been transferred.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR TCR LL_I2C_IsActiveFlag_TCR</li> </ul>                                                       |

**LL\_I2C\_IsActiveFlag\_BERR**

|                                                   |                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_BERR<br/>(I2C_TypeDef * I2Cx)</code>                                                    |
| Function description                              | Indicate the status of Bus error flag.                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• RESET: Clear default value. SET: When a misplaced Start or Stop condition is detected.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR BERR LL_I2C_IsActiveFlag_BERR</li> </ul>                                                      |

**LL\_I2C\_IsActiveFlag\_ARLO**

|                                                   |                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_ARLO<br/>(I2C_TypeDef * I2Cx)</code>                     |
| Function description                              | Indicate the status of Arbitration lost flag.                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• RESET: Clear default value. SET: When arbitration lost.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR ARLO LL_I2C_IsActiveFlag_ARLO</li> </ul>                       |

**LL\_I2C\_IsActiveFlag\_OVR**

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_OVR<br/>(I2C_TypeDef * I2Cx)</code> |
| Function description | Indicate the status of Overrun/Underrun flag (slave mode).                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>         |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>     |

---

|                                             |                                                                                                                                                         |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>• RESET: Clear default value. SET: When an overrun/underrun error occurs (Clock Stretching Disabled).</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• ISR OVR LL_I2C_IsActiveFlag_OVR</li> </ul>                                                                     |

### LL\_I2C\_IsActiveSMBusFlag\_PECERR

|                                             |                                                                                                                                                                                                                                                                                            |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t<br/>LL_I2C_IsActiveSMBusFlag_PECERR (I2C_TypeDef * I2Cx)</code>                                                                                                                                                                                             |
| Function description                        | Indicate the status of SMBus PEC error flag in reception.                                                                                                                                                                                                                                  |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                                                             |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                         |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> <li>• RESET: Clear default value. SET: When the received PEC does not match with the PEC register content.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• ISR PECERR LL_I2C_IsActiveSMBusFlag_PECERR</li> </ul>                                                                                                                                                                                             |

### LL\_I2C\_IsActiveSMBusFlag\_TIMEOUT

|                                             |                                                                                                                                                                                                                                                                         |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t<br/>LL_I2C_IsActiveSMBusFlag_TIMEOUT (I2C_TypeDef * I2Cx)</code>                                                                                                                                                                         |
| Function description                        | Indicate the status of SMBus Timeout detection flag.                                                                                                                                                                                                                    |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                                          |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                      |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> <li>• RESET: Clear default value. SET: When a timeout or extended clock timeout occurs.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• ISR TIMEOUT LL_I2C_IsActiveSMBusFlag_TIMEOUT</li> </ul>                                                                                                                                                                        |

### LL\_I2C\_IsActiveSMBusFlag\_ALERT

|                      |                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t<br/>LL_I2C_IsActiveSMBusFlag_ALERT (I2C_TypeDef * I2Cx)</code>        |
| Function description | Indicate the status of SMBus alert flag.                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                   |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to</li> </ul> |

check whether or not SMBus feature is supported by the I2Cx Instance.

- RESET: Clear default value. SET: When SMBus host configuration, SMBus alert enabled and a falling edge event occurs on SMBA pin.

Reference Manual to  
LL API cross  
reference:

- ISR ALERT LL\_I2C\_IsActiveSMBusFlag\_ALERT

### **LL\_I2C\_IsActiveFlag\_BUSY**

Function name      **`__STATIC_INLINE uint32_t LL_I2C_IsActiveFlag_BUSY(I2C_TypeDef * I2Cx)`**

Function description      Indicate the status of Bus Busy flag.

Parameters      • **I2Cx:** I2C Instance.

Return values      • **State:** of bit (1 or 0).

Notes      • RESET: Clear default value. SET: When a Start condition is detected.

Reference Manual to  
LL API cross  
reference:

- ISR BUSY LL\_I2C\_IsActiveFlag\_BUSY

### **LL\_I2C\_ClearFlag\_ADDR**

Function name      **`__STATIC_INLINE void LL_I2C_ClearFlag_ADDR(I2C_TypeDef * I2Cx)`**

Function description      Clear Address Matched flag.

Parameters      • **I2Cx:** I2C Instance.

Return values      • **None**

Reference Manual to  
LL API cross  
reference:

- ICR ADDRCF LL\_I2C\_ClearFlag\_ADDR

### **LL\_I2C\_ClearFlag\_NACK**

Function name      **`__STATIC_INLINE void LL_I2C_ClearFlag_NACK(I2C_TypeDef * I2Cx)`**

Function description      Clear Not Acknowledge flag.

Parameters      • **I2Cx:** I2C Instance.

Return values      • **None**

Reference Manual to  
LL API cross  
reference:

- ICR NACKCF LL\_I2C\_ClearFlag\_NACK

**LL\_I2C\_ClearFlag\_STOP**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_ClearFlag_STOP (I2C_TypeDef * I2Cx)</code>         |
| Function description                              | Clear Stop detection flag.                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ICR STOPCF LL_I2C_ClearFlag_STOP</li> </ul> |

**LL\_I2C\_ClearFlag\_TXE**

|                                                   |                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_ClearFlag_TXE (I2C_TypeDef * I2Cx)</code>                                                                |
| Function description                              | Clear Transmit data register empty flag (TXE).                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit can be clear by software in order to flush the transmit data register (TXDR).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR TXE LL_I2C_ClearFlag_TXE</li> </ul>                                                           |

**LL\_I2C\_ClearFlag\_BERR**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_ClearFlag_BERR (I2C_TypeDef * I2Cx)</code>         |
| Function description                              | Clear Bus error flag.                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ICR BERRCF LL_I2C_ClearFlag_BERR</li> </ul> |

**LL\_I2C\_ClearFlag\_ARLO**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_ClearFlag_ARLO (I2C_TypeDef * I2Cx)</code>         |
| Function description                              | Clear Arbitration lost flag.                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ICR ARLOCF LL_I2C_ClearFlag_ARLO</li> </ul> |

**LL\_I2C\_ClearFlag\_OVR**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_ClearFlag_OVR (I2C_TypeDef * I2Cx)</code>        |
| Function description                              | Clear Overrun/Underrun flag.                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ICR OVRCF LL_I2C_ClearFlag_OVR</li> </ul> |

**LL\_I2C\_ClearSMBusFlag\_PECERR**

|                                                   |                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_ClearSMBusFlag_PECERR (I2C_TypeDef * I2Cx)</code>                                                                                        |
| Function description                              | Clear SMBus PEC error flag.                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ICR PECCF LL_I2C_ClearSMBusFlag_PECERR</li> </ul>                                                                                 |

**LL\_I2C\_ClearSMBusFlag\_TIMEOUT**

|                                                   |                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_ClearSMBusFlag_TIMEOUT (I2C_TypeDef * I2Cx)</code>                                                                                       |
| Function description                              | Clear SMBus Timeout detection flag.                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ICR TIMEOUTCF LL_I2C_ClearSMBusFlag_TIMEOUT</li> </ul>                                                                            |

**LL\_I2C\_ClearSMBusFlag\_ALERT**

|                      |                                                                                    |
|----------------------|------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_ClearSMBusFlag_ALERT (I2C_TypeDef * I2Cx)</code> |
| Function description | Clear SMBus Alert flag.                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>     |

---

|                                             |                                                                                                                                                                        |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                               | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                                                            |
| Notes                                       | <ul style="list-style-type: none"><li>Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li></ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>ICR ALERTCF LL_I2C_ClearSMBusFlag_ALERT</li></ul>                                                                                |

### LL\_I2C\_EnableAutoEndMode

|                                             |                                                                                                                                                                                                               |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_I2C_EnableAutoEndMode(I2C_TypeDef * I2Cx)</code></b>                                                                                                                          |
| Function description                        | Enable automatic STOP condition generation (master mode).                                                                                                                                                     |
| Parameters                                  | <ul style="list-style-type: none"><li><b>I2Cx:</b> I2C Instance.</li></ul>                                                                                                                                    |
| Return values                               | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                                                                                                   |
| Notes                                       | <ul style="list-style-type: none"><li>Automatic end mode : a STOP condition is automatically sent when NBYTES data are transferred. This bit has no effect in slave mode or when RELOAD bit is set.</li></ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>CR2 AUTOEND LL_I2C_EnableAutoEndMode</li></ul>                                                                                                                          |

### LL\_I2C\_DisableAutoEndMode

|                                             |                                                                                                                                          |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_I2C_DisableAutoEndMode(I2C_TypeDef * I2Cx)</code></b>                                                    |
| Function description                        | Disable automatic STOP condition generation (master mode).                                                                               |
| Parameters                                  | <ul style="list-style-type: none"><li><b>I2Cx:</b> I2C Instance.</li></ul>                                                               |
| Return values                               | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                              |
| Notes                                       | <ul style="list-style-type: none"><li>Software end mode : TC flag is set when NBYTES data are transferred, stretching SCL low.</li></ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>CR2 AUTOEND LL_I2C_DisableAutoEndMode</li></ul>                                                    |

### LL\_I2C\_IsEnabledAutoEndMode

|                                             |                                                                                             |
|---------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_I2C_IsEnabledAutoEndMode(I2C_TypeDef * I2Cx)</code></b> |
| Function description                        | Check if automatic STOP condition is enabled or disabled.                                   |
| Parameters                                  | <ul style="list-style-type: none"><li><b>I2Cx:</b> I2C Instance.</li></ul>                  |
| Return values                               | <ul style="list-style-type: none"><li><b>State:</b> of bit (1 or 0).</li></ul>              |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>CR2 AUTOEND LL_I2C_IsEnabledAutoEndMode</li></ul>     |

**LL\_I2C\_EnableReloadMode**

|                                                   |                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_I2C_EnableReloadMode (I2C_TypeDef * I2Cx)</b>                                                                                         |
| Function description                              | Enable reload mode (master mode).                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>• The transfer is not completed after the NBYTES data transfer, NBYTES will be reloaded when TCR flag is set.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RELOAD LL_I2C_EnableReloadMode</li> </ul>                                                                          |

**LL\_I2C\_DisableReloadMode**

|                                                   |                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_I2C_DisableReloadMode (I2C_TypeDef * I2Cx)</b>                                                                   |
| Function description                              | Disable reload mode (master mode).                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• The transfer is completed after the NBYTES data transfer(STOP or RESTART will follow).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RELOAD LL_I2C_DisableReloadMode</li> </ul>                                                    |

**LL\_I2C\_IsEnabledReloadMode**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE uint32_t LL_I2C_IsEnabledReloadMode (I2C_TypeDef * I2Cx)</b>            |
| Function description                              | Check if reload mode is enabled or disabled.                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RELOAD LL_I2C_IsEnabledReloadMode</li> </ul> |

**LL\_I2C\_SetTransferSize**

|                      |                                                                                                                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE void LL_I2C_SetTransferSize (I2C_TypeDef * I2Cx, uint32_t TransferSize)</b>                                                                                          |
| Function description | Configure the number of bytes for transfer.                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>TransferSize:</b> This parameter must be a value between Min_Data=0x00 and Max_Data=0xFF.</li> </ul> |

---

|                                                   |                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>Changing these bits when START bit is set is not allowed.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR2 NBYTES LL_I2C_SetTransferSize</li> </ul>                         |

### LL\_I2C\_GetTransferSize

|                                                   |                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_GetTransferSize(<br/>I2C_TypeDef * I2Cx)</code>                  |
| Function description                              | Get the number of bytes configured for transfer.                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                           |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x0 and Max_Data=0xFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR2 NBYTES LL_I2C_SetTransferSize</li> </ul>                    |

### LL\_I2C\_AcknowledgeNextData

|                                                   |                                                                                                                                                                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_AcknowledgeNextData(<br/>I2C_TypeDef * I2Cx, uint32_t TypeAcknowledge)</code>                                                                                                                                         |
| Function description                              | Prepare the generation of a ACKnowledge or Non ACKnowledge condition after the address receive match code or next received byte.                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> <li><b>TypeAcknowledge:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_I2C_ACK</li> <li>- LL_I2C_NACK</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>Usage in Slave mode only.</li> </ul>                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR2 NACK LL_I2C_AcknowledgeNextData</li> </ul>                                                                                                                                                                   |

### LL\_I2C\_GenerateStartCondition

|                      |                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_GenerateStartCondition(<br/>I2C_TypeDef * I2Cx)</code>                                                                                        |
| Function description | Generate a START or RESTART condition.                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                   |
| Notes                | <ul style="list-style-type: none"> <li>The START bit can be set even if bus is BUSY or I2C is in slave mode. This action has no effect when RELOAD is set.</li> <li></li> </ul> |

- Reference Manual to  
LL API cross  
reference:
- CR2 START LL\_I2C\_GenerateStartCondition

### **LL\_I2C\_GenerateStopCondition**

|                                                   |                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_I2C_GenerateStopCondition<br/>(I2C_TypeDef * I2Cx)</code></b> |
| Function description                              | Generate a STOP condition after the current byte transfer (master mode).                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 STOP LL_I2C_GenerateStopCondition</li> </ul>     |

### **LL\_I2C\_EnableAuto10BitRead**

|                                                   |                                                                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_I2C_EnableAuto10BitRead<br/>(I2C_TypeDef * I2Cx)</code></b>                                                                                                                                         |
| Function description                              | Enable automatic RESTART Read request condition for 10bit address header (master mode).                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• The master sends the complete 10bit slave address read sequence : Start + 2 bytes 10bit address in Write direction + Restart + first 7 bits of 10bit address in Read direction.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 HEAD10R LL_I2C_EnableAuto10BitRead</li> </ul>                                                                                                                                          |

### **LL\_I2C\_DisableAuto10BitRead**

|                                                   |                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_I2C_DisableAuto10BitRead<br/>(I2C_TypeDef * I2Cx)</code></b>                                   |
| Function description                              | Disable automatic RESTART Read request condition for 10bit address header (master mode).                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                |
| Notes                                             | <ul style="list-style-type: none"> <li>• The master only sends the first 7 bits of 10bit address in Read direction.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 HEAD10R LL_I2C_DisableAuto10BitRead</li> </ul>                                    |

**LL\_I2C\_IsEnabledAuto10BitRead**

|                                                   |                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledAuto10BitRead(I2C_TypeDef * I2Cx)</code>            |
| Function description                              | Check if automatic RESTART Read request condition for 10bit address header is enabled or disabled. |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 HEAD10R LL_I2C_IsEnabledAuto10BitRead</li> </ul>      |

**LL\_I2C\_SetTransferRequest**

|                                                   |                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_SetTransferRequest(I2C_TypeDef * I2Cx, uint32_t TransferRequest)</code>                                                                                                                                                                     |
| Function description                              | Configure the transfer direction (master mode).                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>TransferRequest:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_I2C_REQUEST_WRITE</li> <li>– LL_I2C_REQUEST_READ</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• Changing these bits when START bit is set is not allowed.</li> </ul>                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RD_WRN LL_I2C_SetTransferRequest</li> </ul>                                                                                                                                                                                      |

**LL\_I2C\_GetTransferRequest**

|                                                   |                                                                                                                                                                                                                         |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_GetTransferRequest(I2C_TypeDef * I2Cx)</code>                                                                                                                                     |
| Function description                              | Get the transfer direction requested (master mode).                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_I2C_REQUEST_WRITE</li> <li>– LL_I2C_REQUEST_READ</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RD_WRN LL_I2C_GetTransferRequest</li> </ul>                                                                                                                                |

**LL\_I2C\_SetSlaveAddr**

|                      |                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_SetSlaveAddr(I2C_TypeDef * I2Cx, uint32_t SlaveAddr)</code> |
| Function description | Configure the slave address for transfer (master mode).                                       |

|                                             |                                                                                                                                                                                 |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                  | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> <li><b>SlaveAddr:</b> This parameter must be a value between Min_Data=0x00 and Max_Data=0x3F.</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                   |
| Notes                                       | <ul style="list-style-type: none"> <li>Changing these bits when START bit is set is not allowed.</li> </ul>                                                                     |
| Reference Manual to LL API cross reference: | CR2 SADD LL_I2C_SetSlaveAddr                                                                                                                                                    |

### LL\_I2C\_GetSlaveAddr

|                                             |                                                                                                        |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_I2C_GetSlaveAddr<br/>(I2C_TypeDef * I2Cx)</code>                     |
| Function description                        | Get the slave address programmed for transfer.                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> </ul>                           |
| Return values                               | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x0 and Max_Data=0x3F</li> </ul> |
| Reference Manual to LL API cross reference: | CR2 SADD LL_I2C_GetSlaveAddr                                                                           |

### LL\_I2C\_HandleTransfer

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_HandleTransfer (I2C_TypeDef * I2Cx, uint32_t SlaveAddr, uint32_t SlaveAddrSize, uint32_t TransferSize, uint32_t EndMode, uint32_t Request)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description | Handles I2Cx communication when starting transfer or during transfer (TC or TCR flag are set).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>I2Cx:</b> I2C Instance.</li> <li><b>SlaveAddr:</b> Specifies the slave address to be programmed.</li> <li><b>SlaveAddrSize:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_I2C_ADDRSCLAVE_7BIT</li> <li>- LL_I2C_ADDRSCLAVE_10BIT</li> </ul> </li> <li><b>TransferSize:</b> Specifies the number of bytes to be programmed. This parameter must be a value between Min_Data=0 and Max_Data=255.</li> <li><b>EndMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_I2C_MODE_RELOAD</li> <li>- LL_I2C_MODE_AUTOEND</li> <li>- LL_I2C_MODE_SOFTEND</li> <li>- LL_I2C_MODE_SMBUS_RELOAD</li> <li>- LL_I2C_MODE_SMBUS_AUTOEND_NO_PEC</li> <li>- LL_I2C_MODE_SMBUS_SOFTEND_NO_PEC</li> <li>- LL_I2C_MODE_SMBUS_AUTOEND_WITH_PEC</li> <li>- LL_I2C_MODE_SMBUS_SOFTEND_WITH_PEC</li> </ul> </li> <li><b>Request:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_I2C_GENERATE_NOSTARTSTOP</li> <li>- LL_I2C_GENERATE_STOP</li> </ul> </li> </ul> |

- LL\_I2C\_GENERATE\_START\_READ
- LL\_I2C\_GENERATE\_START\_WRITE
- LL\_I2C\_GENERATE\_RESTART\_7BIT\_READ
- LL\_I2C\_GENERATE\_RESTART\_7BIT\_WRITE
- LL\_I2C\_GENERATE\_RESTART\_10BIT\_READ
- LL\_I2C\_GENERATE\_RESTART\_10BIT\_WRITE

**Return values**

Reference Manual to  
LL API cross  
reference:

- **None**

- CR2 SADD LL\_I2C\_HandleTransfer
- CR2 ADD10 LL\_I2C\_HandleTransfer
- CR2 RD\_WRN LL\_I2C\_HandleTransfer
- CR2 START LL\_I2C\_HandleTransfer
- CR2 STOP LL\_I2C\_HandleTransfer
- CR2 RELOAD LL\_I2C\_HandleTransfer
- CR2 NBYTES LL\_I2C\_HandleTransfer
- CR2 AUTOEND LL\_I2C\_HandleTransfer
- CR2 HEAD10R LL\_I2C\_HandleTransfer

### LL\_I2C\_GetTransferDirection

Function name      **`__STATIC_INLINE uint32_t LL_I2C_GetTransferDirection(I2C_TypeDef * I2Cx)`**

Function description      Indicate the value of transfer direction (slave mode).

Parameters     

- **I2Cx:** I2C Instance.

Return values     

- **Returned:** value can be one of the following values:
  - LL\_I2C\_DIRECTION\_WRITE
  - LL\_I2C\_DIRECTION\_READ

Notes     

- RESET: Write transfer, Slave enters in receiver mode. SET: Read transfer, Slave enters in transmitter mode.

Reference Manual to  
LL API cross  
reference:

- ISR DIR LL\_I2C\_GetTransferDirection

### LL\_I2C\_GetAddressMatchCode

Function name      **`__STATIC_INLINE uint32_t LL_I2C_GetAddressMatchCode(I2C_TypeDef * I2Cx)`**

Function description      Return the slave matched address.

Parameters     

- **I2Cx:** I2C Instance.

Return values     

- **Value:** between Min\_Data=0x00 and Max\_Data=0x3F

Reference Manual to  
LL API cross  
reference:

- ISR ADDCODE LL\_I2C\_GetAddressMatchCode

**LL\_I2C\_EnableSMBusPECCompare**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2C_EnableSMBusPECCompare (I2C_TypeDef * I2Cx)</code>                                                                                                                                                                                                                                                                                                                                                   |
| Function description                              | Enable internal comparison of the SMBus Packet Error byte (transmission or reception mode).                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                                                                                                                                                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> <li>• This feature is cleared by hardware when the PEC byte is transferred, or when a STOP condition or an Address Matched is received. This bit has no effect when RELOAD bit is set. This bit has no effect in device mode when SBC bit is not set.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 PECBYTE LL_I2C_EnableSMBusPECCompare</li> </ul>                                                                                                                                                                                                                                                                                                                                          |

**LL\_I2C\_IsEnabledSMBusPECCompare**

|                                                   |                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2C_IsEnabledSMBusPECCompare (I2C_TypeDef * I2Cx)</code>                                                                                 |
| Function description                              | Check if the SMBus Packet Error byte internal comparison is requested or not.                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 PECBYTE LL_I2C_IsEnabledSMBusPECCompare</li> </ul>                                                                            |

**LL\_I2C\_GetSMBusPEC**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_I2C_GetSMBusPEC (I2C_TypeDef * I2Cx)</code>                                                                                              |
| Function description | Get the SMBus Packet Error byte calculated.                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xFF</li> </ul>                                                                  |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_SMBUS_ALL_INSTANCE(I2Cx) can be used to check whether or not SMBus feature is supported by the I2Cx Instance.</li> </ul> |

- Reference Manual to LL API cross reference:
- PECR PEC LL\_I2C\_GetSMBusPEC

### **LL\_I2C\_ReceiveData8**

|                      |                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint8_t LL_I2C_ReceiveData8 (I2C_TypeDef * I2Cx)</code>                             |
| Function description | Read Receive Data register.                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                            |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xFF</li> </ul> |

- Reference Manual to LL API cross reference:
- RXDR RXDATA LL\_I2C\_ReceiveData8

### **LL\_I2C\_TransmitData8**

|                      |                                                                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2C_TransmitData8 (I2C_TypeDef * I2Cx, uint8_t Data)</code>                                                            |
| Function description | Write in Transmit Data Register .                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>Data:</b> Value between Min_Data=0x00 and Max_Data=0xFF</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                      |

- Reference Manual to LL API cross reference:
- TXDR TXDATA LL\_I2C\_TransmitData8

### **LL\_I2C\_Init**

|                      |                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t LL_I2C_Init (I2C_TypeDef * I2Cx, LL_I2C_InitTypeDef * I2C_InitStruct)</code>                                                                                                                                |
| Function description | Initialize the I2C registers according to the specified parameters in I2C_InitStruct.                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> <li>• <b>I2C_InitStruct:</b> pointer to a LL_I2C_InitTypeDef structure.</li> </ul>                                                                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: I2C registers are initialized</li> <li>– ERROR: Not applicable</li> </ul> </li> </ul> |

### **LL\_I2C\_DeInit**

|                      |                                                                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t LL_I2C_DeInit (I2C_TypeDef * I2Cx)</code>                                                                                                                                     |
| Function description | De-initialize the I2C registers to their default reset values.                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> I2C Instance.</li> </ul>                                                                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: I2C registers are de-initialized</li> </ul> </li> </ul> |

- ERROR: I2C registers are not de-initialized

### **LL\_I2C\_StructInit**

|                      |                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_I2C_StructInit (LL_I2C_InitTypeDef * I2C_InitStruct)</b>                                                   |
| Function description | Set each LL_I2C_InitTypeDef field to default value.                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2C_InitStruct:</b> Pointer to a LL_I2C_InitTypeDef structure.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                       |

## **69.3 I2C Firmware driver defines**

### **69.3.1 I2C**

#### ***Master Addressing Mode***

**LL\_I2C\_ADDRESSING\_MODE\_7BIT** Master operates in 7-bit addressing mode.

**LL\_I2C\_ADDRESSING\_MODE\_10BIT** Master operates in 10-bit addressing mode.

#### ***Slave Address Length***

**LL\_I2C\_ADDRSOLVE\_7BIT** Slave Address in 7-bit.

**LL\_I2C\_ADDRSOLVE\_10BIT** Slave Address in 10-bit.

#### ***Analog Filter Selection***

**LL\_I2C\_ANALOGFILTER\_ENABLE** Analog filter is enabled.

**LL\_I2C\_ANALOGFILTER\_DISABLE** Analog filter is disabled.

#### ***Clear Flags Defines***

**LL\_I2C\_ICR\_ADDRCF** Address Matched flag

**LL\_I2C\_ICR\_NACKCF** Not Acknowledge flag

**LL\_I2C\_ICR\_STOPCF** Stop detection flag

**LL\_I2C\_ICR\_BERRCF** Bus error flag

**LL\_I2C\_ICR\_ARLOCF** Arbitration Lost flag

**LL\_I2C\_ICR\_OVRCF** Overrun/Underrun flag

**LL\_I2C\_ICR\_PECCF** PEC error flag

**LL\_I2C\_ICR\_TIMOUTCF** Timeout detection flag

**LL\_I2C\_ICR\_ALERTCF** Alert flag

#### ***Read Write Direction***

**LL\_I2C\_DIRECTION\_WRITE** Write transfer request by master, slave enters receiver mode.

**LL\_I2C\_DIRECTION\_READ** Read transfer request by master, slave enters transmitter mode.

#### ***DMA Register Data***

**LL\_I2C\_DMA\_REG\_DATA\_TRANSMIT** Get address of data register used for transmission

|                             |                                                 |
|-----------------------------|-------------------------------------------------|
| LL_I2C_DMA_REG_DATA_RECEIVE | Get address of data register used for reception |
|-----------------------------|-------------------------------------------------|

#### **Start And Stop Generation**

|                                     |                                                          |
|-------------------------------------|----------------------------------------------------------|
| LL_I2C_GENERATE_NOSTARTSTOP         | Don't Generate Stop and Start condition.                 |
| LL_I2C_GENERATE_STOP                | Generate Stop condition (Size should be set to 0).       |
| LL_I2C_GENERATE_START_READ          | Generate Start for read request.                         |
| LL_I2C_GENERATE_START_WRITE         | Generate Start for write request.                        |
| LL_I2C_GENERATE_RESTART_7BIT_READ   | Generate Restart for read request, slave 7Bit address.   |
| LL_I2C_GENERATE_RESTART_7BIT_WRITE  | Generate Restart for write request, slave 7Bit address.  |
| LL_I2C_GENERATE_RESTART_10BIT_READ  | Generate Restart for read request, slave 10Bit address.  |
| LL_I2C_GENERATE_RESTART_10BIT_WRITE | Generate Restart for write request, slave 10Bit address. |

#### **Get Flags Defines**

|                    |                                     |
|--------------------|-------------------------------------|
| LL_I2C_ISR_TXE     | Transmit data register empty        |
| LL_I2C_ISR_RXIS    | Transmit interrupt status           |
| LL_I2C_ISR_RXNE    | Receive data register not empty     |
| LL_I2C_ISR_ADDR    | Address matched (slave mode)        |
| LL_I2C_ISR_NACKF   | Not Acknowledge received flag       |
| LL_I2C_ISR_STOPF   | Stop detection flag                 |
| LL_I2C_ISR_TC      | Transfer Complete (master mode)     |
| LL_I2C_ISR_TCR     | Transfer Complete Reload            |
| LL_I2C_ISR_BERR    | Bus error                           |
| LL_I2C_ISR_ARLO    | Arbitration lost                    |
| LL_I2C_ISR_OVR     | Overrun/Underrun (slave mode)       |
| LL_I2C_ISR_PECERR  | PEC Error in reception (SMBus mode) |
| LL_I2C_ISR_TIMEOUT | Timeout detection flag (SMBus mode) |
| LL_I2C_ISR_ALERT   | SMBus alert (SMBus mode)            |
| LL_I2C_ISR_BUSY    | Bus busy                            |

#### **Acknowledge Generation**

|             |                                           |
|-------------|-------------------------------------------|
| LL_I2C_ACK  | ACK is sent after current received byte.  |
| LL_I2C_NACK | NACK is sent after current received byte. |

#### **IT Defines**

|                 |                     |
|-----------------|---------------------|
| LL_I2C_CR1_TXIE | TX Interrupt enable |
| LL_I2C_CR1_RXIE | RX Interrupt enable |

|                   |                                             |
|-------------------|---------------------------------------------|
| LL_I2C_CR1_ADDRIE | Address match Interrupt enable (slave only) |
| LL_I2C_CR1_NACKIE | Not acknowledge received Interrupt enable   |
| LL_I2C_CR1_STOPIE | STOP detection Interrupt enable             |
| LL_I2C_CR1_TCIE   | Transfer Complete interrupt enable          |
| LL_I2C_CR1_ERRIE  | Error interrupts enable                     |

#### **Transfer End Mode**

|                                    |                                                          |
|------------------------------------|----------------------------------------------------------|
| LL_I2C_MODE_RELOAD                 | Enable I2C Reload mode.                                  |
| LL_I2C_MODE_AUTOEND                | Enable I2C Automatic end mode with no HW PEC comparison. |
| LL_I2C_MODE_SOFTEND                | Enable I2C Software end mode with no HW PEC comparison.  |
| LL_I2C_MODE_SMBUS_RELOAD           | Enable SMBUS Automatic end mode with HW PEC comparison.  |
| LL_I2C_MODE_SMBUS_AUTOEND_NO_PEC   | Enable SMBUS Automatic end mode with HW PEC comparison.  |
| LL_I2C_MODE_SMBUS_SOFTEND_NO_PEC   | Enable SMBUS Software end mode with HW PEC comparison.   |
| LL_I2C_MODE_SMBUS_AUTOEND_WITH_PEC | Enable SMBUS Automatic end mode with HW PEC comparison.  |
| LL_I2C_MODE_SMBUS_SOFTEND_WITH_PEC | Enable SMBUS Software end mode with HW PEC comparison.   |

#### **Own Address 1 Length**

|                          |                                    |
|--------------------------|------------------------------------|
| LL_I2C_OWNADDRESS1_7BIT  | Own address 1 is a 7-bit address.  |
| LL_I2C_OWNADDRESS1_10BIT | Own address 1 is a 10-bit address. |

#### **Own Address 2 Masks**

|                           |                                                       |
|---------------------------|-------------------------------------------------------|
| LL_I2C_OWNADDRESS2_NOMASK | Own Address2 No mask.                                 |
| LL_I2C_OWNADDRESS2_MASK01 | Only Address2 bits[7:2] are compared.                 |
| LL_I2C_OWNADDRESS2_MASK02 | Only Address2 bits[7:3] are compared.                 |
| LL_I2C_OWNADDRESS2_MASK03 | Only Address2 bits[7:4] are compared.                 |
| LL_I2C_OWNADDRESS2_MASK04 | Only Address2 bits[7:5] are compared.                 |
| LL_I2C_OWNADDRESS2_MASK05 | Only Address2 bits[7:6] are compared.                 |
| LL_I2C_OWNADDRESS2_MASK06 | Only Address2 bits[7] are compared.                   |
| LL_I2C_OWNADDRESS2_MASK07 | No comparison is done. All Address2 are acknowledged. |

#### **Peripheral Mode**

|                          |                                                             |
|--------------------------|-------------------------------------------------------------|
| LL_I2C_MODE_I2C          | I2C Master or Slave mode                                    |
| LL_I2C_MODE_SMBUS_HOST   | SMBus Host address acknowledge                              |
| LL_I2C_MODE_SMBUS_DEVICE | SMBus Device default mode (Default address not acknowledge) |

**LL\_I2C\_MODE\_SMBUS\_DEVICE\_ARP** SMBus Device Default address acknowledge

#### **Transfer Request Direction**

**LL\_I2C\_REQUEST\_WRITE** Master request a write transfer.

**LL\_I2C\_REQUEST\_READ** Master request a read transfer.

#### **SMBus TimeoutA Mode SCL SDA Timeout**

**LL\_I2C\_SMBUS\_TIMEOUTA\_MODE\_SCL\_LOW** TimeoutA is used to detect SCL low level timeout.

**LL\_I2C\_SMBUS\_TIMEOUTA\_MODE\_SDA\_SCL\_HIGH** TimeoutA is used to detect both SCL and SDA high level timeout.

#### **SMBus Timeout Selection**

**LL\_I2C\_SMBUS\_TIMEOUTA** TimeoutA enable bit

**LL\_I2C\_SMBUS\_TIMEOUTB** TimeoutB (extended clock) enable bit

**LL\_I2C\_SMBUS\_ALL\_TIMEOUT** TimeoutA and TimeoutB (extended clock) enable bits

#### **Convert SDA SCL timings**

**\_LL\_I2C\_CONVERT\_TIMINGS** **Description:**

- Configure the SDA setup, hold time and the SCL high, low period.

#### **Parameters:**

- \_PRESCALER\_**: This parameter must be a value between Min\_Data=0 and Max\_Data=0xF.
- \_DATA\_SETUP\_TIME\_**: This parameter must be a value between Min\_Data=0 and Max\_Data=0xF. ( $tscldel = (SCLDEL+1)xtpresc$ )
- \_DATA\_HOLD\_TIME\_**: This parameter must be a value between Min\_Data=0 and Max\_Data=0xF. ( $tsdadel = SDADELxtpresc$ )
- \_CLOCK\_HIGH\_PERIOD\_**: This parameter must be a value between Min\_Data=0 and Max\_Data=0xFF. ( $tsclh = (SCLH+1)xtpresc$ )
- \_CLOCK\_LOW\_PERIOD\_**: This parameter must be a value between Min\_Data=0 and Max\_Data=0xFF. ( $tscll = (SCLL+1)xtpresc$ )

#### **Return value:**

- Value: between Min\_Data=0 and Max\_Data=0xFFFFFFFF

#### **Common Write and read registers Macros**

**LL\_I2C\_WriteReg** **Description:**

- Write a value in I2C register.

#### **Parameters:**

- \_INSTANCE\_**: I2C Instance
- \_REG\_**: Register to be written
- \_VALUE\_**: Value to be written in the register

**Return value:**

- None

LL\_I2C\_ReadReg

**Description:**

- Read a value in I2C register.

**Parameters:**

- \_\_INSTANCE\_\_: I2C Instance
- \_\_REG\_\_: Register to be read

**Return value:**

- Register: value

## 70 LL I2S Generic Driver

### 70.1 I2S Firmware driver registers structures

#### 70.1.1 LL\_I2S\_InitTypeDef

##### Data Fields

- *uint32\_t Mode*
- *uint32\_t Standard*
- *uint32\_t DataFormat*
- *uint32\_t MCLKOutput*
- *uint32\_t AudioFreq*
- *uint32\_t ClockPolarity*

##### Field Documentation

- ***uint32\_t LL\_I2S\_InitTypeDef::Mode***  
Specifies the I2S operating mode. This parameter can be a value of **I2S\_LL\_EC\_MODE**This feature can be modified afterwards using unitary function **LL\_I2S\_SetTransferMode()**.
- ***uint32\_t LL\_I2S\_InitTypeDef::Standard***  
Specifies the standard used for the I2S communication. This parameter can be a value of **I2S\_LL\_EC\_STANDARD**This feature can be modified afterwards using unitary function **LL\_I2S\_SetStandard()**.
- ***uint32\_t LL\_I2S\_InitTypeDef::DataFormat***  
Specifies the data format for the I2S communication. This parameter can be a value of **I2S\_LL\_EC\_DATA\_FORMAT**This feature can be modified afterwards using unitary function **LL\_I2S\_SetDataFormat()**.
- ***uint32\_t LL\_I2S\_InitTypeDef::MCLKOutput***  
Specifies whether the I2S MCLK output is enabled or not. This parameter can be a value of **I2S\_LL\_EC\_MCLK\_OUTPUT**This feature can be modified afterwards using unitary functions **LL\_I2S\_EnableMasterClock()** or **LL\_I2S\_DisableMasterClock**.
- ***uint32\_t LL\_I2S\_InitTypeDef::AudioFreq***  
Specifies the frequency selected for the I2S communication. This parameter can be a value of **I2S\_LL\_EC\_AUDIO\_FREQ**Audio Frequency can be modified afterwards using Reference manual formulas to calculate Prescaler Linear, Parity and unitary functions **LL\_I2S\_SetPrescalerLinear()** and **LL\_I2S\_SetPrescalerParity()** to set it.
- ***uint32\_t LL\_I2S\_InitTypeDef::ClockPolarity***  
Specifies the idle state of the I2S clock. This parameter can be a value of **I2S\_LL\_EC\_POLARITY**This feature can be modified afterwards using unitary function **LL\_I2S\_SetClockPolarity()**.

### 70.2 I2S Firmware driver API description

#### 70.2.1 Detailed description of functions

##### LL\_I2S\_Enable

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <b>__STATIC_INLINE void LL_I2S_Enable (SPI_TypeDef * SPIx)</b>                |
| Function description | Select I2S mode and Enable I2S peripheral.                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul> |

|                                             |                                                                                                                    |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>I2SCFGR I2SMOD LL_I2S_Enable</li> <li>I2SCFGR I2SE LL_I2S_Enable</li> </ul> |

### LL\_I2S\_Disable

|                                             |                                                                               |
|---------------------------------------------|-------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_I2S_Disable (SPI_TypeDef * SPIx)</code></b>   |
| Function description                        | Disable I2S peripheral.                                                       |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>   |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                 |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>I2SCFGR I2SE LL_I2S_Disable</li> </ul> |

### LL\_I2S\_IsEnabled

|                                             |                                                                                   |
|---------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_I2S_IsEnabled (SPI_TypeDef * SPIx)</code></b> |
| Function description                        | Check if I2S peripheral is enabled.                                               |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>       |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>I2SCFGR I2SE LL_I2S_IsEnabled</li> </ul>   |

### LL\_I2S\_SetDataFormat

|                                             |                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_I2S_SetDataFormat (SPI_TypeDef * SPIx, uint32_t DataFormat)</code></b>                                                                                                                                                                                                                                    |
| Function description                        | Set I2S data frame length.                                                                                                                                                                                                                                                                                                                |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> <li><b>DataFormat:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_I2S_DATAFORMAT_16B</li> <li>LL_I2S_DATAFORMAT_16B_EXTENDED</li> <li>LL_I2S_DATAFORMAT_24B</li> <li>LL_I2S_DATAFORMAT_32B</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                             |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>I2SCFGR DATLEN LL_I2S_SetDataFormat</li> <li>I2SCFGR CHLEN LL_I2S_SetDataFormat</li> </ul>                                                                                                                                                                                                         |

**LL\_I2S\_GetDataFormat**

|                                                   |                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_I2S_GetDataFormat(SPI_TypeDef * SPIx)</code>                                                                                                                                                                                                                                 |
| Function description                              | Get I2S data frame length.                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_I2S_DATAFORMAT_16B</li> <li>– LL_I2S_DATAFORMAT_16B_EXTENDED</li> <li>– LL_I2S_DATAFORMAT_24B</li> <li>– LL_I2S_DATAFORMAT_32B</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• I2SCFGR DATLEN LL_I2S_GetDataFormat</li> <li>• I2SCFGR CHLEN LL_I2S_GetDataFormat</li> </ul>                                                                                                                                                                         |

**LL\_I2S\_SetClockPolarity**

|                                                   |                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_I2S_SetClockPolarity(SPI_TypeDef * SPIx, uint32_t ClockPolarity)</code>                                                                                                                                                                             |
| Function description                              | Set I2S clock polarity.                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> <li><b>ClockPolarity:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_I2S_POLARITY_LOW</li> <li>– LL_I2S_POLARITY_HIGH</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• I2SCFGR CKPOL LL_I2S_SetClockPolarity</li> </ul>                                                                                                                                                                                        |

**LL\_I2S\_GetClockPolarity**

|                                                   |                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_I2S_GetClockPolarity(SPI_TypeDef * SPIx)</code>                                                                                                                                                |
| Function description                              | Get I2S clock polarity.                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_I2S_POLARITY_LOW</li> <li>– LL_I2S_POLARITY_HIGH</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• I2SCFGR CKPOL LL_I2S_GetClockPolarity</li> </ul>                                                                                                                                       |

**LL\_I2S\_SetStandard**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2S_SetStandard (SPI_TypeDef * SPIx, uint32_t Standard)</code>                                                                                                                                                                                                                                                                                     |
| Function description                              | Set I2S standard protocol.                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>Standard:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_I2S_STANDARD_PHILIPS</li> <li>- LL_I2S_STANDARD_MSB</li> <li>- LL_I2S_STANDARD_LSB</li> <li>- LL_I2S_STANDARD_PCM_SHORT</li> <li>- LL_I2S_STANDARD_PCM_LONG</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• I2SCFGR I2SSTD LL_I2S_SetStandard</li> <li>• I2SCFGR PCMSYNC LL_I2S_SetStandard</li> </ul>                                                                                                                                                                                                                                              |

**LL\_I2S\_GetStandard**

|                                                   |                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2S_GetStandard (SPI_TypeDef * SPIx)</code>                                                                                                                                                                                                                                                      |
| Function description                              | Get I2S standard protocol.                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_I2S_STANDARD_PHILIPS</li> <li>- LL_I2S_STANDARD_MSB</li> <li>- LL_I2S_STANDARD_LSB</li> <li>- LL_I2S_STANDARD_PCM_SHORT</li> <li>- LL_I2S_STANDARD_PCM_LONG</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• I2SCFGR I2SSTD LL_I2S_SetStandard</li> <li>• I2SCFGR PCMSYNC LL_I2S_SetStandard</li> </ul>                                                                                                                                                                                                |

**LL\_I2S\_SetTransferMode**

|                      |                                                                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_I2S_SetTransferMode (SPI_TypeDef * SPIx, uint32_t Mode)</code>                                                                                                                                                                                                                                         |
| Function description | Set I2S transfer mode.                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>Mode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_I2S_MODE_SLAVE_TX</li> <li>- LL_I2S_MODE_SLAVE_RX</li> <li>- LL_I2S_MODE_MASTER_TX</li> <li>- LL_I2S_MODE_MASTER_RX</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                      |

- Reference Manual to  
LL API cross  
reference:
- I2SCFGR I2SCFG LL\_I2S\_SetTransferMode

### **LL\_I2S\_GetTransferMode**

Function name      **`_STATIC_INLINE uint32_t LL_I2S_GetTransferMode(SPI_TypeDef * SPIx)`**

Function description      Get I2S transfer mode.

Parameters      • **SPIx:** SPI Instance

Return values      • **Returned:** value can be one of the following values:

- LL\_I2S\_MODE\_SLAVE\_TX
- LL\_I2S\_MODE\_SLAVE\_RX
- LL\_I2S\_MODE\_MASTER\_TX
- LL\_I2S\_MODE\_MASTER\_RX

- Reference Manual to  
LL API cross  
reference:
- I2SCFGR I2SCFG LL\_I2S\_SetTransferMode

### **LL\_I2S\_SetPrescalerLinear**

Function name      **`_STATIC_INLINE void LL_I2S_SetPrescalerLinear(SPI_TypeDef * SPIx, uint8_t PrescalerLinear)`**

Function description      Set I2S linear prescaler.

Parameters      • **SPIx:** SPI Instance

• **PrescalerLinear:** Value between Min\_Data=0x02 and Max\_Data=0xFF

Return values      • **None**

- Reference Manual to  
LL API cross  
reference:
- I2SPR I2SDIV LL\_I2S\_SetPrescalerLinear

### **LL\_I2S\_GetPrescalerLinear**

Function name      **`_STATIC_INLINE uint32_t LL_I2S_GetPrescalerLinear(SPI_TypeDef * SPIx)`**

Function description      Get I2S linear prescaler.

Parameters      • **SPIx:** SPI Instance

Return values      • **PrescalerLinear:** Value between Min\_Data=0x02 and Max\_Data=0xFF

- Reference Manual to  
LL API cross  
reference:
- I2SPR I2SDIV LL\_I2S\_SetPrescalerLinear

**LL\_I2S\_SetPrescalerParity**

|                                                   |                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_I2S_SetPrescalerParity(SPI_TypeDef * SPIx, uint32_t PrescalerParity)</code>                                                                                                                                                                                     |
| Function description                              | Set I2S parity prescaler.                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>PrescalerParity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_I2S_PRESCALER_PARITY EVEN</li> <li>– LL_I2S_PRESCALER_PARITY ODD</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• I2SPR ODD LL_I2S_SetPrescalerParity</li> </ul>                                                                                                                                                                                                      |

**LL\_I2S\_GetPrescalerParity**

|                                                   |                                                                                                                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_I2S_GetPrescalerParity(SPI_TypeDef * SPIx)</code>                                                                                                                                                      |
| Function description                              | Get I2S parity prescaler.                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_I2S_PRESCALER_PARITY EVEN</li> <li>– LL_I2S_PRESCALER_PARITY ODD</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• I2SPR ODD LL_I2S_GetPrescalerParity</li> </ul>                                                                                                                                                 |

**LL\_I2S\_EnableMasterClock**

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_I2S_EnableMasterClock(SPI_TypeDef * SPIx)</code>            |
| Function description                              | Enable the master clock output (Pin MCK)                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• I2SPR MCKOE LL_I2S_EnableMasterClock</li> </ul> |

**LL\_I2S\_DisableMasterClock**

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_I2S_DisableMasterClock(SPI_TypeDef * SPIx)</code> |
| Function description | Disable the master clock output (Pin MCK)                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                |

- Reference Manual to I2SPR MCKOE LL\_I2S\_DisableMasterClock  
LL API cross reference:
- I2SPR MCKOE LL\_I2S\_DisableMasterClock

### **LL\_I2S\_IsEnabledMasterClock**

|                      |                                                                                      |
|----------------------|--------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_I2S_IsEnabledMasterClock(SPI_TypeDef * SPIx)</code> |
| Function description | Check if the master clock output (Pin MCK) is enabled.                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>   |

Reference Manual to I2SPR MCKOE LL\_I2S\_IsEnabledMasterClock  
LL API cross reference:

### **LL\_I2S\_IsActiveFlag\_RXNE**

|                      |                                                                                    |
|----------------------|------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_I2S_IsActiveFlag_RXNE(SPI_TypeDef * SPIx)</code>  |
| Function description | Check if Rx buffer is not empty.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |

Reference Manual to SR RXNE LL\_I2S\_IsActiveFlag\_RXNE  
LL API cross reference:

### **LL\_I2S\_IsActiveFlag\_TXE**

|                      |                                                                                    |
|----------------------|------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_I2S_IsActiveFlag_TXE(SPI_TypeDef * SPIx)</code>   |
| Function description | Check if Tx buffer is empty.                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |

Reference Manual to SR TXE LL\_I2S\_IsActiveFlag\_TXE  
LL API cross reference:

### **LL\_I2S\_IsActiveFlag\_BSY**

|                      |                                                                                    |
|----------------------|------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_I2S_IsActiveFlag_BSY(SPI_TypeDef * SPIx)</code>   |
| Function description | Get busy flag.                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |

- Reference Manual to  
LL API cross  
reference:
- SR BSY LL\_I2S\_IsActiveFlag\_BSY

### **LL\_I2S\_IsActiveFlag\_OVR**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_I2S_IsActiveFlag_OVR(SPI_TypeDef * SPIx)</code>   |
| Function description                              | Get overrun error flag.                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR OVR LL_I2S_IsActiveFlag_OVR</li> </ul> |

### **LL\_I2S\_IsActiveFlag\_UDR**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_I2S_IsActiveFlag_UDR(SPI_TypeDef * SPIx)</code>   |
| Function description                              | Get underrun error flag.                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR UDR LL_I2S_IsActiveFlag_UDR</li> </ul> |

### **LL\_I2S\_IsActiveFlag\_FRE**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_I2S_IsActiveFlag_FRE(SPI_TypeDef * SPIx)</code>   |
| Function description                              | Get frame format error flag.                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR FRE LL_I2S_IsActiveFlag_FRE</li> </ul> |

### **LL\_I2S\_IsActiveFlag\_CHSIDE**

|                      |                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_I2S_IsActiveFlag_CHSIDE(SPI_TypeDef * SPIx)</code>                                                                                           |
| Function description | Get channel side flag.                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                 |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                            |
| Notes                | <ul style="list-style-type: none"> <li>• 0: Channel Left has to be transmitted or has been received 1: Channel Right has to be transmitted or has been received It</li> </ul> |

has no significance in PCM mode.

Reference Manual to  
LL API cross  
reference:

- SR CHSIDE LL\_I2S\_IsActiveFlag\_CHSIDE

### **LL\_I2S\_ClearFlag\_OVR**

Function name **\_STATIC\_INLINE void LL\_I2S\_ClearFlag\_OVR (SPI\_TypeDef \* SPIx)**

Function description Clear overrun error flag.

Parameters • **SPIx:** SPI Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
• SR OVR LL\_I2S\_ClearFlag\_OVR

### **LL\_I2S\_ClearFlag\_UDR**

Function name **\_STATIC\_INLINE void LL\_I2S\_ClearFlag\_UDR (SPI\_TypeDef \* SPIx)**

Function description Clear underrun error flag.

Parameters • **SPIx:** SPI Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
• SR UDR LL\_I2S\_ClearFlag\_UDR

### **LL\_I2S\_ClearFlag\_FRE**

Function name **\_STATIC\_INLINE void LL\_I2S\_ClearFlag\_FRE (SPI\_TypeDef \* SPIx)**

Function description Clear frame format error flag.

Parameters • **SPIx:** SPI Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
• SR FRE LL\_I2S\_ClearFlag\_FRE

### **LL\_I2S\_EnableIT\_ERR**

Function name **\_STATIC\_INLINE void LL\_I2S\_EnableIT\_ERR (SPI\_TypeDef \* SPIx)**

Function description Enable error IT.

Parameters • **SPIx:** SPI Instance

Return values • **None**

---

|                                             |                                                                                                                                                                 |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"><li>This bit controls the generation of an interrupt when an error condition occurs (OVR, UDR and FRE in I2S mode).</li></ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>CR2 ERRIE LL_I2S_EnableIT_ERR</li></ul>                                                                                   |

### LL\_I2S\_EnableIT\_RXNE

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_I2S_EnableIT_RXNE (SPI_TypeDef * SPIx)</code></b> |
| Function description | Enable Rx buffer not empty IT.                                                    |
| Parameters           | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>         |
| Return values        | <ul style="list-style-type: none"><li><b>None</b></li></ul>                       |

Reference Manual to LL API cross reference:

- CR2 RXNEIE LL\_I2S\_EnableIT\_RXNE

### LL\_I2S\_EnableIT\_TXE

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_I2S_EnableIT_TXE (SPI_TypeDef * SPIx)</code></b> |
| Function description | Enable Tx buffer empty IT.                                                       |
| Parameters           | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>        |
| Return values        | <ul style="list-style-type: none"><li><b>None</b></li></ul>                      |

Reference Manual to LL API cross reference:

- CR2 TXEIE LL\_I2S\_EnableIT\_TXE

### LL\_I2S\_DisableIT\_ERR

|                                             |                                                                                                                                                                 |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_I2S_DisableIT_ERR (SPI_TypeDef * SPIx)</code></b>                                                                               |
| Function description                        | Disable error IT.                                                                                                                                               |
| Parameters                                  | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>                                                                                       |
| Return values                               | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                                                     |
| Notes                                       | <ul style="list-style-type: none"><li>This bit controls the generation of an interrupt when an error condition occurs (OVR, UDR and FRE in I2S mode).</li></ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>CR2 ERRIE LL_I2S_DisableIT_ERR</li></ul>                                                                                  |

**LL\_I2S\_DisableIT\_RXNE**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2S_DisableIT_RXNE (SPI_TypeDef * SPIx)</code>         |
| Function description                              | Disable Rx buffer not empty IT.                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RXNEIE LL_I2S_DisableIT_RXNE</li> </ul> |

**LL\_I2S\_DisableIT\_TXE**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2S_DisableIT_TXE (SPI_TypeDef * SPIx)</code>        |
| Function description                              | Disable Tx buffer empty IT.                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 TXEIE LL_I2S_DisableIT_TXE</li> </ul> |

**LL\_I2S\_IsEnabledIT\_ERR**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2S_IsEnabledIT_ERR (SPI_TypeDef * SPIx)</code>    |
| Function description                              | Check if ERR IT is enabled.                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 ERRIE LL_I2S_IsEnabledIT_ERR</li> </ul> |

**LL\_I2S\_IsEnabledIT\_RXNE**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2S_IsEnabledIT_RXNE (SPI_TypeDef * SPIx)</code>     |
| Function description                              | Check if RXNE IT is enabled.                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RXNEIE LL_I2S_IsEnabledIT_RXNE</li> </ul> |

**LL\_I2S\_IsEnabledIT\_TXE**

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2S_IsEnabledIT_TXE(SPI_TypeDef * SPIx)</code> |
| Function description                              | Check if TXE IT is enabled.                                                      |
| Parameters                                        | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>        |
| Return values                                     | <ul style="list-style-type: none"><li><b>State:</b> of bit (1 or 0).</li></ul>   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR2 TXEIE LL_I2S_IsEnabledIT_TXE</li></ul> |

**LL\_I2S\_EnableDMAReq\_RX**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2S_EnableDMAReq_RX(SPI_TypeDef * SPIx)</code>       |
| Function description                              | Enable DMA Rx.                                                                     |
| Parameters                                        | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>          |
| Return values                                     | <ul style="list-style-type: none"><li><b>None</b></li></ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR2 RXDMAEN LL_I2S_EnableDMAReq_RX</li></ul> |

**LL\_I2S\_DisableDMAReq\_RX**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2S_DisableDMAReq_RX(SPI_TypeDef * SPIx)</code>       |
| Function description                              | Disable DMA Rx.                                                                     |
| Parameters                                        | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>           |
| Return values                                     | <ul style="list-style-type: none"><li><b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR2 RXDMAEN LL_I2S_DisableDMAReq_RX</li></ul> |

**LL\_I2S\_IsEnabledDMAReq\_RX**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2S_IsEnabledDMAReq_RX(SPI_TypeDef * SPIx)</code>   |
| Function description                              | Check if DMA Rx is enabled.                                                           |
| Parameters                                        | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li><b>State:</b> of bit (1 or 0).</li></ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR2 RXDMAEN LL_I2S_IsEnabledDMAReq_RX</li></ul> |

**LL\_I2S\_EnableDMAReq\_TX**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2S_EnableDMAReq_TX<br/>(SPI_TypeDef * SPIx)</code>      |
| Function description                              | Enable DMA Tx.                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 TXDMAEN LL_I2S_EnableDMAReq_TX</li> </ul> |

**LL\_I2S\_DisableDMAReq\_TX**

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2S_DisableDMAReq_TX<br/>(SPI_TypeDef * SPIx)</code>      |
| Function description                              | Disable DMA Tx.                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 TXDMAEN LL_I2S_DisableDMAReq_TX</li> </ul> |

**LL\_I2S\_IsEnabledDMAReq\_TX**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_I2S_IsEnabledDMAReq_TX<br/>(SPI_TypeDef * SPIx)</code>  |
| Function description                              | Check if DMA Tx is enabled.                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 TXDMAEN LL_I2S_IsEnabledDMAReq_TX</li> </ul> |

**LL\_I2S\_ReceiveData16**

|                                                   |                                                                                                                      |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint16_t LL_I2S_ReceiveData16<br/>(SPI_TypeDef * SPIx)</code>                                  |
| Function description                              | Read 16-Bits in data register.                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>RxDATA:</b> Value between Min_Data=0x0000 and Max_Data=0xFFFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DR DR LL_I2S_ReceiveData16</li> </ul>                                       |

**LL\_I2S\_TransmitData16**

|                                                   |                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_I2S_TransmitData16 (SPI_TypeDef * SPIx, uint16_t TxData)</code>                                                             |
| Function description                              | Write 16-Bits in data register.                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>TxData:</b> Value between Min_Data=0x0000 and Max_Data=0xFFFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DR DR LL_I2S_TransmitData16</li> </ul>                                                                           |

**LL\_I2S\_DelInit**

|                      |                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_I2S_DelInit (SPI_TypeDef * SPIx)</code>                                                                                                                                                                                        |
| Function description | De-initialize the SPI/I2S registers to their default reset values.                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: SPI registers are de-initialized</li> <li>– ERROR: SPI registers are not de-initialized</li> </ul> </li> </ul> |

**LL\_I2S\_Init**

|                      |                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_I2S_Init (SPI_TypeDef * SPIx, LL_I2S_InitTypeDef * I2S_InitStruct)</code>                                                                                                                                                                               |
| Function description | Initializes the SPI/I2S registers according to the specified parameters in I2S_InitStruct.                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>I2S_InitStruct:</b> pointer to a LL_I2S_InitTypeDef structure</li> </ul>                                                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: SPI registers are Initialized</li> <li>– ERROR: SPI registers are not Initialized</li> </ul> </li> </ul>                                |
| Notes                | <ul style="list-style-type: none"> <li>• As some bits in SPI configuration registers can only be written when the SPI is disabled (SPI_CR1_SPE bit =0), SPI IP should be in disabled state prior calling this function. Otherwise, ERROR result will be returned.</li> </ul> |

**LL\_I2S\_StructInit**

|                      |                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void LL_I2S_StructInit (LL_I2S_InitTypeDef * I2S_InitStruct)</code>                                                                                        |
| Function description | Set each LL_I2S_InitTypeDef field to default value.                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2S_InitStruct:</b> pointer to a LL_I2S_InitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                  |

**LL\_I2S\_ConfigPrescaler**

|                      |                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void LL_I2S_ConfigPrescaler (SPI_TypeDef * SPIx, uint32_t PrescalerLinear, uint32_t PrescalerParity)</code>                                                                                                                                                                                                                                                       |
| Function description | Set linear and parity prescaler.                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>PrescalerLinear:</b> value: Min_Data=0x02 and Max_Data=0xFF.</li> <li>• <b>PrescalerParity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_I2S_PRESCALER_PARITY_EVEN</li> <li>– LL_I2S_PRESCALER_PARITY_ODD</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                         |
| Notes                | <ul style="list-style-type: none"> <li>• To calculate value of PrescalerLinear(I2SDIV[7:0] bits) and PrescalerParity(ODD bit) Check Audio frequency table and formulas inside Reference Manual (SPI/I2S).</li> </ul>                                                                                                                                                    |

**LL\_I2S\_InitFullDuplex**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_I2S_InitFullDuplex (SPI_TypeDef * I2Sxext, LL_I2S_InitTypeDef * I2S_InitStruct)</code>                                                                                                                                                                                                                                                                                                            |
| Function description | Configures the full duplex mode for the I2Sx peripheral using its extension I2Sxext according to the specified parameters in the I2S_InitStruct.                                                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2Sxext:</b> SPI Instance</li> <li>• <b>I2S_InitStruct:</b> pointer to a LL_I2S_InitTypeDef structure</li> </ul>                                                                                                                                                                                                                                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: I2Sxext registers are Initialized</li> <li>– ERROR: I2Sxext registers are not Initialized</li> </ul> </li> </ul>                                                                                                                                                                  |
| Notes                | <ul style="list-style-type: none"> <li>• The structure pointed by I2S_InitStruct parameter should be the same used for the master I2S peripheral. In this case, if the master is configured as transmitter, the slave will be receiver and vice versa. Or you can force a different mode by modifying the field I2S_Mode to the value I2S_SlaveRx or I2S_SlaveTx independently of the master configuration.</li> </ul> |

## 70.3 I2S Firmware driver defines

### 70.3.1 I2S

***Audio Frequency***

|                                    |                                         |
|------------------------------------|-----------------------------------------|
| <code>LL_I2S_AUDIOFREQ_192K</code> | Audio Frequency configuration 192000 Hz |
| <code>LL_I2S_AUDIOFREQ_96K</code>  | Audio Frequency configuration 96000 Hz  |
| <code>LL_I2S_AUDIOFREQ_48K</code>  | Audio Frequency configuration 48000 Hz  |
| <code>LL_I2S_AUDIOFREQ_44K</code>  | Audio Frequency configuration 44100 Hz  |
| <code>LL_I2S_AUDIOFREQ_32K</code>  | Audio Frequency configuration 32000 Hz  |
| <code>LL_I2S_AUDIOFREQ_22K</code>  | Audio Frequency configuration 22050 Hz  |

|                          |                                               |
|--------------------------|-----------------------------------------------|
| LL_I2S_AUDIOFREQ_16K     | Audio Frequency configuration 16000 Hz        |
| LL_I2S_AUDIOFREQ_11K     | Audio Frequency configuration 11025 Hz        |
| LL_I2S_AUDIOFREQ_8K      | Audio Frequency configuration 8000 Hz         |
| LL_I2S_AUDIOFREQ_DEFAULT | Audio Freq not specified. Register I2SDIV = 2 |

**Data format**

|                                |                                           |
|--------------------------------|-------------------------------------------|
| LL_I2S_DATAFORMAT_16B          | Data length 16 bits, Channel lenght 16bit |
| LL_I2S_DATAFORMAT_16B_EXTENDED | Data length 16 bits, Channel lenght 32bit |
| LL_I2S_DATAFORMAT_24B          | Data length 24 bits, Channel lenght 32bit |
| LL_I2S_DATAFORMAT_32B          | Data length 16 bits, Channel lenght 32bit |

**Get Flags Defines**

|                |                                 |
|----------------|---------------------------------|
| LL_I2S_SR_RXNE | Rx buffer not empty flag        |
| LL_I2S_SR_TXE  | Tx buffer empty flag            |
| LL_I2S_SR_BSY  | Busy flag                       |
| LL_I2S_SR_UDR  | Underrun flag                   |
| LL_I2S_SR_OVR  | Overrun flag                    |
| LL_I2S_SR_FRE  | TI mode frame format error flag |

**MCLK Output**

|                            |                                 |
|----------------------------|---------------------------------|
| LL_I2S_MCLK_OUTPUT_DISABLE | Master clock output is disabled |
| LL_I2S_MCLK_OUTPUT_ENABLE  | Master clock output is enabled  |

**Operation Mode**

|                       |                         |
|-----------------------|-------------------------|
| LL_I2S_MODE_SLAVE_TX  | Slave Tx configuration  |
| LL_I2S_MODE_SLAVE_RX  | Slave Rx configuration  |
| LL_I2S_MODE_MASTER_TX | Master Tx configuration |
| LL_I2S_MODE_MASTER_RX | Master Rx configuration |

**Clock Polarity**

|                      |                                  |
|----------------------|----------------------------------|
| LL_I2S_POLARITY_LOW  | Clock steady state is low level  |
| LL_I2S_POLARITY_HIGH | Clock steady state is high level |

**Prescaler Factor**

|                              |                                                    |
|------------------------------|----------------------------------------------------|
| LL_I2S_PRESCALER_PARITY_EVEN | Odd factor: Real divider value is = I2SDIV * 2     |
| LL_I2S_PRESCALER_PARITY_ODD  | Odd factor: Real divider value is = (I2SDIV * 2)+1 |

**I2s Standard**

|                           |                                           |
|---------------------------|-------------------------------------------|
| LL_I2S_STANDARD_PHILIPS   | I2S standard philips                      |
| LL_I2S_STANDARD_MSB       | MSB justified standard (left justified)   |
| LL_I2S_STANDARD_LSB       | LSB justified standard (right justified)  |
| LL_I2S_STANDARD_PCM_SHORT | PCM standard, short frame synchronization |
| LL_I2S_STANDARD_PCM_LONG  | PCM standard, long frame synchronization  |

***Common Write and read registers Macros*****LL\_I2S\_WriteReg****Description:**

- Write a value in I2S register.

**Parameters:**

- \_\_INSTANCE\_\_: I2S Instance
- \_\_REG\_\_: Register to be written
- \_\_VALUE\_\_: Value to be written in the register

**Return value:**

- None

**LL\_I2S\_ReadReg****Description:**

- Read a value in I2S register.

**Parameters:**

- \_\_INSTANCE\_\_: I2S Instance
- \_\_REG\_\_: Register to be read

**Return value:**

- Register: value

## 71 LL IWDG Generic Driver

### 71.1 IWDG Firmware driver API description

#### 71.1.1 Detailed description of functions

##### LL\_IWDG\_Enable

Function name **`__STATIC_INLINE void LL_IWDG_Enable (IWDG_TypeDef * IWDGx)`**

Function description Start the Independent Watchdog.

Parameters • **IWDGx:** IWDG Instance

Return values • **None**

Notes • Except if the hardware watchdog option is selected

Reference Manual to

LL API cross

reference:

**`__STATIC_INLINE void LL_IWDG_ReloadCounter (IWDG_TypeDef * IWDGx)`**

Function description Reloads IWDG counter with value defined in the reload register.

Parameters • **IWDGx:** IWDG Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:

##### LL\_IWDG\_EnableWriteAccess

Function name **`__STATIC_INLINE void LL_IWDG_EnableWriteAccess (IWDG_TypeDef * IWDGx)`**

Function description Enable write access to IWDG\_PR, IWDG\_RLR and IWDG\_WINR registers.

Parameters • **IWDGx:** IWDG Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:

**LL\_IWDG\_DisableWriteAccess**

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_IWDG_DisableWriteAccess<br/>(IWDG_TypeDef * IWDGx)</code> |
| Function description                              | Disable write access to IWDG_PR, IWDG_RLR and IWDG_WINR registers.                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>IWDGx:</b> IWDG Instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• KR KEY LL_IWDG_DisableWriteAccess</li> </ul>   |

**LL\_IWDG\_SetPrescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_IWDG_SetPrescaler<br/>(IWDG_TypeDef * IWDGx, uint32_t Prescaler)</code>                                                                                                                                                                                                                                                                                                                                     |
| Function description                              | Select the prescaler of the IWDG.                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>IWDGx:</b> IWDG Instance</li> <li>• <b>Prescaler:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_IWDG_PRESCALER_4</li> <li>- LL_IWDG_PRESCALER_8</li> <li>- LL_IWDG_PRESCALER_16</li> <li>- LL_IWDG_PRESCALER_32</li> <li>- LL_IWDG_PRESCALER_64</li> <li>- LL_IWDG_PRESCALER_128</li> <li>- LL_IWDG_PRESCALER_256</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• PR PR LL_IWDG_SetPrescaler</li> </ul>                                                                                                                                                                                                                                                                                                                                                            |

**LL\_IWDG\_GetPrescaler**

|                                     |                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                       | <code>__STATIC_INLINE uint32_t LL_IWDG_GetPrescaler<br/>(IWDG_TypeDef * IWDGx)</code>                                                                                                                                                                                                                                                                                                    |
| Function description                | Get the selected prescaler of the IWDG.                                                                                                                                                                                                                                                                                                                                                  |
| Parameters                          | <ul style="list-style-type: none"> <li>• <b>IWDGx:</b> IWDG Instance</li> </ul>                                                                                                                                                                                                                                                                                                          |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_IWDG_PRESCALER_4</li> <li>- LL_IWDG_PRESCALER_8</li> <li>- LL_IWDG_PRESCALER_16</li> <li>- LL_IWDG_PRESCALER_32</li> <li>- LL_IWDG_PRESCALER_64</li> <li>- LL_IWDG_PRESCALER_128</li> <li>- LL_IWDG_PRESCALER_256</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• PR PR LL_IWDG_GetPrescaler</li> </ul>                                                                                                                                                                                                                                                                                                           |

reference:

### **LL\_IWDG\_SetReloadCounter**

|                                             |                                                                                                                                                         |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><u>STATIC_INLINE void LL_IWDG_SetReloadCounter (IWDG_TypeDef * IWDGx, uint32_t Counter)</u></b>                                                      |
| Function description                        | Specify the IWDG down-counter reload value.                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>IWDGx:</b> IWDG Instance</li> <li>• <b>Counter:</b> Value between Min_Data=0 and Max_Data=0xFFFF</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                         |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• RLR RL LL_IWDG_SetReloadCounter</li> </ul>                                                                     |

### **LL\_IWDG\_GetReloadCounter**

|                                             |                                                                                                          |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------|
| Function name                               | <b><u>STATIC_INLINE uint32_t LL_IWDG_GetReloadCounter (IWDG_TypeDef * IWDGx)</u></b>                     |
| Function description                        | Get the specified IWDG down-counter reload value.                                                        |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>IWDGx:</b> IWDG Instance</li> </ul>                          |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0 and Max_Data=0xFFFF</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• RLR RL LL_IWDG_GetReloadCounter</li> </ul>                      |

### **LL\_IWDG\_SetWindow**

|                                             |                                                                                                                                                        |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><u>STATIC_INLINE void LL_IWDG_SetWindow (IWDG_TypeDef * IWDGx, uint32_t Window)</u></b>                                                             |
| Function description                        | Specify high limit of the window value to be compared to the down-counter.                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>IWDGx:</b> IWDG Instance</li> <li>• <b>Window:</b> Value between Min_Data=0 and Max_Data=0xFFFF</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• WINR WIN LL_IWDG_SetWindow</li> </ul>                                                                         |

### **LL\_IWDG\_GetWindow**

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <b><u>STATIC_INLINE uint32_t LL_IWDG_GetWindow (IWDG_TypeDef * IWDGx)</u></b>   |
| Function description | Get the high limit of the window value specified.                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>IWDGx:</b> IWDG Instance</li> </ul> |

|                                                   |                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0 and Max_Data=0x0FF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>WINR WIN LL_IWDG_GetWindow</li> </ul>                          |

### LL\_IWDG\_IsActiveFlag\_PVU

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_IWDG_IsActiveFlag_PVU<br/>(IWDG_TypeDef * IWDGx)</code> |
| Function description                              | Check if flag Prescaler Value Update is set or not.                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>IWDGx:</b> IWDG Instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SR PVU LL_IWDG_IsActiveFlag_PVU</li> </ul>         |

### LL\_IWDG\_IsActiveFlag\_RVU

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_IWDG_IsActiveFlag_RVU<br/>(IWDG_TypeDef * IWDGx)</code> |
| Function description                              | Check if flag Reload Value Update is set or not.                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>IWDGx:</b> IWDG Instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SR RVU LL_IWDG_IsActiveFlag_RVU</li> </ul>         |

### LL\_IWDG\_IsActiveFlag\_WVU

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_IWDG_IsActiveFlag_WVU<br/>(IWDG_TypeDef * IWDGx)</code> |
| Function description                              | Check if flag Window Value Update is set or not.                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>IWDGx:</b> IWDG Instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SR WVU LL_IWDG_IsActiveFlag_WVU</li> </ul>         |

### LL\_IWDG\_IsReady

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_IWDG_IsReady<br/>(IWDG_TypeDef * IWDGx)</code> |
| Function description | Check if all flags Prescaler, Reload & Window Value Update are reset or not.     |
| Parameters           | <ul style="list-style-type: none"> <li><b>IWDGx:</b> IWDG Instance</li> </ul>    |

- |                                                                    |                                                                                                                                                                                           |
|--------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values<br>Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <b>State:</b> of bits (1 or 0).</li> <li>• SR PVU LL_IWDG_IsReady</li> <li>• SR WVU LL_IWDG_IsReady</li> <li>• SR RVU LL_IWDG_IsReady</li> </ul> |
|--------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## 71.2 IWDG Firmware driver defines

### 71.2.1 IWDG

#### *Get Flags Defines*

|                |                                      |
|----------------|--------------------------------------|
| LL_IWDG_SR_PVU | Watchdog prescaler value update      |
| LL_IWDG_SR_RVU | Watchdog counter reload value update |
| LL_IWDG_SR_WVU | Watchdog counter window value update |

#### *Prescaler Divider*

|                       |                |
|-----------------------|----------------|
| LL_IWDG_PRESCALER_4   | Divider by 4   |
| LL_IWDG_PRESCALER_8   | Divider by 8   |
| LL_IWDG_PRESCALER_16  | Divider by 16  |
| LL_IWDG_PRESCALER_32  | Divider by 32  |
| LL_IWDG_PRESCALER_64  | Divider by 64  |
| LL_IWDG_PRESCALER_128 | Divider by 128 |
| LL_IWDG_PRESCALER_256 | Divider by 256 |

#### *Common Write and read registers Macros*

##### `LL_IWDG_WriteReg`    **Description:**

- Write a value in IWDG register.

##### **Parameters:**

- `__INSTANCE__`: IWDG Instance
- `__REG__`: Register to be written
- `__VALUE__`: Value to be written in the register

##### **Return value:**

- None

##### `LL_IWDG_ReadReg`    **Description:**

- Read a value in IWDG register.

##### **Parameters:**

- `__INSTANCE__`: IWDG Instance
- `__REG__`: Register to be read

##### **Return value:**

- Register: value

## 72 LL OPAMP Generic Driver

### 72.1 OPAMP Firmware driver registers structures

#### 72.1.1 LL\_OPAMP\_InitTypeDef

##### Data Fields

- *uint32\_t FunctionalMode*
- *uint32\_t InputNonInverting*
- *uint32\_t InputInverting*

##### Field Documentation

- ***uint32\_t LL\_OPAMP\_InitTypeDef::FunctionalMode***

Set OPAMP functional mode by setting internal connections: OPAMP operation in standalone, follower, ... This parameter can be a value of

***OPAMP\_LL\_EC\_FUNCTIONAL\_MODE***

**Note:**If OPAMP is configured in mode PGA, the gain can be configured using function **LL\_OPAMP\_SetPGAGain()**. This feature can be modified afterwards using unitary function **LL\_OPAMP\_SetFunctionalMode()**.

- ***uint32\_t LL\_OPAMP\_InitTypeDef::InputNonInverting***

Set OPAMP input non-inverting connection. This parameter can be a value of ***OPAMP\_LL\_EC\_INPUT\_NONINVERTING***This feature can be modified afterwards using unitary function **LL\_OPAMP\_SetInputNonInverting()**.

- ***uint32\_t LL\_OPAMP\_InitTypeDef::InputInverting***

Set OPAMP inverting input connection. This parameter can be a value of ***OPAMP\_LL\_EC\_INPUT\_INVERTING***

**Note:**OPAMP inverting input is used with OPAMP in mode standalone or PGA with external capacitors for filtering circuit. Otherwise (OPAMP in mode follower), OPAMP inverting input is not used (not connected to GPIO pin), this parameter is discarded. This feature can be modified afterwards using unitary function **LL\_OPAMP\_SetInputInverting()**.

### 72.2 OPAMP Firmware driver API description

#### 72.2.1 Detailed description of functions

##### LL\_OPAMP\_SetMode

Function name      ***\_STATIC\_INLINE void LL\_OPAMP\_SetMode  
(OPAMP\_TypeDef \* OPAMPx, uint32\_t Mode)***

Function description      Set OPAMP mode calibration or functional.

Parameters     
 

- **OPAMPx:** OPAMP instance
- **Mode:** This parameter can be one of the following values:
  - **LL\_OPAMP\_MODE\_FUNCTIONAL**
  - **LL\_OPAMP\_MODE\_CALIBRATION**

Return values     
 

- **None**

Notes     
 

- OPAMP mode corresponds to functional or calibration mode: functional mode: OPAMP operation in standalone, follower, ... Set functional mode using function

LL\_OPAMP\_SetFunctionalMode().calibration mode: offset calibration of the selected transistors differential pair NMOS or PMOS.

Reference Manual to  
LL API cross  
reference:

- CSR CALON LL\_OPAMP\_SetMode

### LL\_OPAMP\_GetMode

Function name

`_STATIC_INLINE uint32_t LL_OPAMP_GetMode  
(OPAMP_TypeDef * OPAMPx)`

Function description

Get OPAMP mode calibration or functional.

Parameters

- OPAMPx:** OPAMP instance

Return values

- Returned:** value can be one of the following values:
  - LL\_OPAMP\_MODE\_FUNCTIONAL
  - LL\_OPAMP\_MODE\_CALIBRATION

Notes

- OPAMP mode corresponds to functional or calibration mode:  
functional mode: OPAMP operation in standalone, follower, ...  
Set functional mode using function  
LL\_OPAMP\_SetFunctionalMode().calibration mode: offset  
calibration of the selected transistors differential pair NMOS  
or PMOS.

Reference Manual to  
LL API cross  
reference:

- CSR CALON LL\_OPAMP\_GetMode

### LL\_OPAMP\_SetFunctionalMode

Function name

`_STATIC_INLINE void LL_OPAMP_SetFunctionalMode  
(OPAMP_TypeDef * OPAMPx, uint32_t FunctionalMode)`

Function description

Set OPAMP functional mode by setting internal connections.

Parameters

- OPAMPx:** OPAMP instance
- FunctionalMode:** This parameter can be one of the following values:
  - LL\_OPAMP\_MODE\_STANDALONE
  - LL\_OPAMP\_MODE\_FOLLOWER
  - LL\_OPAMP\_MODE\_PGA
  - LL\_OPAMP\_MODE\_PGA\_EXT\_FILT\_IO0
  - LL\_OPAMP\_MODE\_PGA\_EXT\_FILT\_IO1

Return values

- None**

Notes

- This function reset bit of calibration mode to ensure to be in functional mode, in order to have OPAMP parameters (inputs selection, ...) set with the corresponding OPAMP mode to be effective.

Reference Manual to  
LL API cross  
reference:

- CSR VMSEL LL\_OPAMP\_SetFunctionalMode

**LL\_OPAMP\_GetFunctionalMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_OPAMP_GetFunctionalMode<br/>(OPAMP_TypeDef * OPAMPx)</code>                                                                                                                                                                                                                                                              |
| Function description                              | Get OPAMP functional mode from setting of internal connections.                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                                                                                                                                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_OPAMP_MODE_STANDALONE</li> <li>- LL_OPAMP_MODE_FOLLOWER</li> <li>- LL_OPAMP_MODE_PGA</li> <li>- LL_OPAMP_MODE_PGA_EXT_FILT_IO0</li> <li>- LL_OPAMP_MODE_PGA_EXT_FILT_IO1</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR VMSEL LL_OPAMP_SetFunctionalMode</li> </ul>                                                                                                                                                                                                                                                                  |

**LL\_OPAMP\_SetPGAGain**

|                                                   |                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_OPAMP_SetPGAGain<br/>(OPAMP_TypeDef * OPAMPx, uint32_t PGAGain)</code>                                                                                                                                                                                                                                              |
| Function description                              | Set OPAMP PGA gain.                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> <li>• <b>PGAGain:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_OPAMP_PGA_GAIN_2</li> <li>- LL_OPAMP_PGA_GAIN_4</li> <li>- LL_OPAMP_PGA_GAIN_8</li> <li>- LL_OPAMP_PGA_GAIN_16</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• Preliminarily, OPAMP must be set in mode PGA using function LL_OPAMP_SetFunctionalMode().</li> </ul>                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR PGGAIN LL_OPAMP_SetPGAGain</li> </ul>                                                                                                                                                                                                                                                               |

**LL\_OPAMP\_GetPGAGain**

|                      |                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_OPAMP_GetPGAGain<br/>(OPAMP_TypeDef * OPAMPx)</code>                                                                                                                                                                                                           |
| Function description | Get OPAMP PGA gain.                                                                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                                                                                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_OPAMP_PGA_GAIN_2</li> <li>- LL_OPAMP_PGA_GAIN_4</li> <li>- LL_OPAMP_PGA_GAIN_8</li> <li>- LL_OPAMP_PGA_GAIN_16</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• Preliminarily, OPAMP must be set in mode PGA using</li> </ul>                                                                                                                                                                                          |

Reference Manual to  
LL API cross  
reference:

`function LL_OPAMP_SetFunctionalMode();`

- CSR PGGAIN LL\_OPAMP\_GetPGAGain

### **LL\_OPAMP\_SetInputNonInverting**

Function name

`__STATIC_INLINE void LL_OPAMP_SetInputNonInverting(  
 OPAMP_TypeDef * OPAMPx, uint32_t InputNonInverting)`

Function description

Set OPAMP non-inverting input connection.

Parameters

- **OPAMPx:** OPAMP instance
- **InputNonInverting:** This parameter can be one of the following values: (1) Parameter specific to OPAMP instances: OPAMP4.
  - LL\_OPAMP\_INPUT\_NONINVERT\_IO0
  - LL\_OPAMP\_INPUT\_NONINVERT\_IO1
  - LL\_OPAMP\_INPUT\_NONINVERT\_IO2
  - LL\_OPAMP\_INPUT\_NONINVERT\_IO3
  - LL\_OPAMP\_INPUT\_NONINV\_DAC1\_CH1 (1)
  - LL\_OPAMP\_INPUT\_NONINV\_DAC1\_CH2 (2)
- (2) Parameter specific to OPAMP instances: OPAMP1, OPAMP3.

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CSR VPSEL LL\_OPAMP\_SetInputNonInverting

### **LL\_OPAMP\_GetInputNonInverting**

Function name

`__STATIC_INLINE uint32_t LL_OPAMP_GetInputNonInverting(  
 OPAMP_TypeDef * OPAMPx)`

Function description

Get OPAMP non-inverting input connection.

Parameters

- **OPAMPx:** OPAMP instance

Return values

- **Returned:** value can be one of the following values: (1) Parameter specific to OPAMP instances: OPAMP4.
  - LL\_OPAMP\_INPUT\_NONINVERT\_IO0
  - LL\_OPAMP\_INPUT\_NONINVERT\_IO1
  - LL\_OPAMP\_INPUT\_NONINVERT\_IO2
  - LL\_OPAMP\_INPUT\_NONINVERT\_IO3
  - LL\_OPAMP\_INPUT\_NONINV\_DAC1\_CH1 (1)
  - LL\_OPAMP\_INPUT\_NONINV\_DAC1\_CH2 (2)
- (2) Parameter specific to OPAMP instances: OPAMP1, OPAMP3.

Reference Manual to  
LL API cross  
reference:

- CSR VPSEL LL\_OPAMP\_GetInputNonInverting

**LL\_OPAMP\_SetInputInverting**

|                                                   |                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_OPAMP_SetInputInverting<br/>(OPAMP_TypeDef * OPAMPx, uint32_t InputInverting)</code>                                                                                                                                                                                                                     |
| Function description                              | Set OPAMP inverting input connection.                                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> <li>• <b>InputInverting:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_OPAMP_INPUT_INVERT_IO0</li> <li>– LL_OPAMP_INPUT_INVERT_IO1</li> <li>– LL_OPAMP_INPUT_INVERT_CONNECT_NO</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• OPAMP inverting input is used with OPAMP in mode standalone or PGA with external capacitors for filtering circuit. Otherwise (OPAMP in mode follower), OPAMP inverting input is not used (not connected to GPIO pin).</li> </ul>                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR VMSEL LL_OPAMP_SetInputInverting</li> </ul>                                                                                                                                                                                                                                               |

**LL\_OPAMP\_GetInputInverting**

|                                                   |                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_OPAMP_GetInputInverting<br/>(OPAMP_TypeDef * OPAMPx)</code>                                                                                                                                                                                  |
| Function description                              | Get OPAMP inverting input connection.                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                                                                                                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_OPAMP_INPUT_INVERT_IO0</li> <li>– LL_OPAMP_INPUT_INVERT_IO1</li> <li>– LL_OPAMP_INPUT_INVERT_CONNECT_NO</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR VMSEL LL_OPAMP_GetInputInverting</li> </ul>                                                                                                                                                                                       |

**LL\_OPAMP\_SetInputNonInvertingSecondary**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void<br/>LL_OPAMP_SetInputNonInvertingSecondary<br/>(OPAMP_TypeDef * OPAMPx, uint32_t InputNonInverting)</code>                                                                                                                                                                                                                                                                                                                                                                    |
| Function description | Set OPAMP non-inverting input secondary connection.                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> <li>• <b>InputNonInverting:</b> This parameter can be one of the following values: (1) Parameter specific to OPAMP instances: OPAMP4. <ul style="list-style-type: none"> <li>– LL_OPAMP_INPUT_NONINVERT_IO0_SEC</li> <li>– LL_OPAMP_INPUT_NONINVERT_IO1_SEC</li> <li>– LL_OPAMP_INPUT_NONINVERT_IO2_SEC</li> <li>– LL_OPAMP_INPUT_NONINVERT_IO3_SEC</li> <li>– LL_OPAMP_INPUT_NONINV_DAC1_CH1_SEC (1)</li> </ul> </li> </ul> |

- LL\_OPAMP\_INPUT\_NONINV\_DAC1\_CH2\_SEC (2)
- (2) Parameter specific to OPAMP instances: OPAMP1, OPAMP3.

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CSR VPSSEL LL\_OPAMP\_SetInputNonInvertingSecondary

**LL\_OPAMP\_GetInputNonInvertingSecondary**

Function name

```
__STATIC_INLINE uint32_t
LL_OPAMP_GetInputNonInvertingSecondary
(OPAMP_TypeDef * OPAMPx)
```

Function description

Get OPAMP non-inverting input secondary connection.

Parameters

- **OPAMPx:** OPAMP instance

Return values

- **Returned:** value can be one of the following values: (1) Parameter specific to OPAMP instances: OPAMP4.

- LL\_OPAMP\_INPUT\_NONINVERT\_IO0\_SEC
- LL\_OPAMP\_INPUT\_NONINVERT\_IO1\_SEC
- LL\_OPAMP\_INPUT\_NONINVERT\_IO2\_SEC
- LL\_OPAMP\_INPUT\_NONINVERT\_IO3\_SEC
- LL\_OPAMP\_INPUT\_NONINV\_DAC1\_CH1\_SEC (1)
- LL\_OPAMP\_INPUT\_NONINV\_DAC1\_CH2\_SEC (2)

- (2) Parameter specific to OPAMP instances: OPAMP1, OPAMP3.

Reference Manual to  
LL API cross  
reference:

- CSR VPSSEL LL\_OPAMP\_GetInputNonInvertingSecondary

**LL\_OPAMP\_SetInputInvertingSecondary**

Function name

```
__STATIC_INLINE void
LL_OPAMP_SetInputInvertingSecondary (OPAMP_TypeDef * OPAMPx, uint32_t InputInverting)
```

Function description

Set OPAMP inverting input secondary connection.

Parameters

- **OPAMPx:** OPAMP instance
- **InputInverting:** This parameter can be one of the following values:
  - LL\_OPAMP\_INPUT\_INVERT\_IO0\_SEC
  - LL\_OPAMP\_INPUT\_INVERT\_IO1\_SEC

Return values

- **None**

Notes

- OPAMP inverting input is used with OPAMP in mode standalone or PGA with external capacitors for filtering circuit. Otherwise (OPAMP in mode follower), OPAMP inverting input is not used (not connected to GPIO pin).

Reference Manual to  
LL API cross  
reference:

- CSR VMSSEL LL\_OPAMP\_SetInputInvertingSecondary

**LL\_OPAMP\_GetInputInvertingSecondary**

|                                                   |                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_OPAMP_GetInputInvertingSecondary (OPAMP_TypeDef *<br/>OPAMPx)</code>                                                                                                                                           |
| Function description                              | Get OPAMP inverting input secondary connection.                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_OPAMP_INPUT_INVERT_IO0_SEC</li> <li>– LL_OPAMP_INPUT_INVERT_IO1_SEC</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR VMSSEL LL_OPAMP_GetInputInvertingSecondary</li> </ul>                                                                                                                                                   |

**LL\_OPAMP\_SetInputsMuxMode**

|                                                   |                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_OPAMP_SetInputsMuxMode<br/>(OPAMP_TypeDef * OPAMPx, uint32_t InputsMuxMode)</code>                                                                                                                                                                                       |
| Function description                              | Set OPAMP inputs multiplexer mode.                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> <li>• <b>InputsMuxMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_OPAMP_INPUT_MUX_DISABLE</li> <li>– LL_OPAMP_INPUT_MUX_TIM1_CH6</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR TCMEN LL_OPAMP_SetInputsMuxMode</li> </ul>                                                                                                                                                                                                                |

**LL\_OPAMP\_GetInputsMuxMode**

|                                                   |                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_OPAMP_GetInputsMuxMode<br/>(OPAMP_TypeDef * OPAMPx)</code>                                                                                                                                                    |
| Function description                              | Get OPAMP inputs multiplexer mode.                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_OPAMP_INPUT_MUX_DISABLE</li> <li>– LL_OPAMP_INPUT_MUX_TIM1_CH6</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR TCMEN LL_OPAMP_SetInputsMuxMode</li> </ul>                                                                                                                                                         |

**LL\_OPAMP\_SetTrimmingMode**

|                      |                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_OPAMP_SetTrimmingMode<br/>(OPAMP_TypeDef * OPAMPx, uint32_t TrimmingMode)</code> |
| Function description | Set OPAMP trimming mode.                                                                                       |

|                                                   |                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> <li>• <b>TrimmingMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_OPAMP_TRIMMING_FACTORY</li> <li>– LL_OPAMP_TRIMMING_USER</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR USERTRIM LL_OPAMP_SetTrimmingMode</li> </ul>                                                                                                                                                                                                       |

### LL\_OPAMP\_GetTrimmingMode

|                                                   |                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_OPAMP_GetTrimmingMode(<br/>OPAMP_TypeDef * OPAMPx)</code>                                                                                                                                                |
| Function description                              | Get OPAMP trimming mode.                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_OPAMP_TRIMMING_FACTORY</li> <li>– LL_OPAMP_TRIMMING_USER</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR USERTRIM LL_OPAMP_GetTrimmingMode</li> </ul>                                                                                                                                                 |

### LL\_OPAMP\_SetCalibrationSelection

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_OPAMP_SetCalibrationSelection(<br/>OPAMP_TypeDef * OPAMPx, uint32_t TransistorsDiffPair)</code>                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Set OPAMP offset to calibrate the selected transistors differential pair NMOS or PMOS.                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> <li>• <b>TransistorsDiffPair:</b> This parameter can be one of the following values: (1) Default parameters to be used for calibration using two trimming steps (one with each transistors differential pair NMOS and PMOS)           <ul style="list-style-type: none"> <li>– LL_OPAMP_TRIMMING_NMOS (1)</li> <li>– LL_OPAMP_TRIMMING_PMOS (1)</li> <li>– LL_OPAMP_TRIMMING_NMOS_VREF_50PC_VDDA</li> <li>– LL_OPAMP_TRIMMING_PMOS_VREF_3_3PC_VDDA</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• Preliminarily, OPAMP must be set in mode calibration using function LL_OPAMP_SetMode().</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual<br>to LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR CALSEL LL_OPAMP_SetCalibrationSelection</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

**LL\_OPAMP\_GetCalibrationSelection**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE uint32_t LL_OPAMP_GetCalibrationSelection (OPAMP_TypeDef * OPAMPx)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description                        | Get OPAMP offset to calibrate the selected transistors differential pair NMOS or PMOS.                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (1) Default parameters to be used for calibration using two trimming steps (one with each transistors differential pair NMOS and PMOS) <ul style="list-style-type: none"> <li>– <code>LL_OPAMP_TRIMMING_NMOS</code> (1)</li> <li>– <code>LL_OPAMP_TRIMMING_PMOS</code> (1)</li> <li>– <code>LL_OPAMP_TRIMMING_NMOS_VREF_50PC_VDDA</code></li> <li>– <code>LL_OPAMP_TRIMMING_PMOS_VREF_3_3PC_VDDA</code></li> </ul> </li> </ul> |
| Notes                                       | <ul style="list-style-type: none"> <li>• Preliminarily, OPAMP must be set in mode calibration using function <code>LL_OPAMP_SetMode()</code>.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR CALSEL <code>LL_OPAMP_GetCalibrationSelection</code></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                       |

**LL\_OPAMP\_SetCalibrationVrefOutput**

|                                             |                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE void LL_OPAMP_SetCalibrationVrefOutput (OPAMP_TypeDef * OPAMPx, uint32_t CalibrationVrefOutput)</code></b>                                                                                                                                                                                              |
| Function description                        | Set OPAMP calibration internal reference voltage to output.                                                                                                                                                                                                                                                                      |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> <li>• <b>CalibrationVrefOutput:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_OPAMP_VREF_OUTPUT_DISABLE</code></li> <li>– <code>LL_OPAMP_VREF_OUTPUT_ENABLE</code></li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CSR TSTREF <code>LL_OPAMP_SetCalibrationVrefOutput</code></li> </ul>                                                                                                                                                                                                                    |

**LL\_OPAMP\_GetCalibrationVrefOutput**

|                                  |                                                                                                                                                                                                                                                                   |
|----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                    | <b><code>__STATIC_INLINE uint32_t LL_OPAMP_GetCalibrationVrefOutput (OPAMP_TypeDef * OPAMPx)</code></b>                                                                                                                                                           |
| Function description             | Get OPAMP calibration internal reference voltage to output.                                                                                                                                                                                                       |
| Parameters                       | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                                                                                                 |
| Return values                    | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_OPAMP_VREF_OUTPUT_DISABLE</code></li> <li>– <code>LL_OPAMP_VREF_OUTPUT_ENABLE</code></li> </ul> </li> </ul> |
| Reference Manual to LL API cross | <ul style="list-style-type: none"> <li>• CSR TSTREF <code>LL_OPAMP_GetCalibrationVrefOutput</code></li> </ul>                                                                                                                                                     |

reference:

### **LL\_OPAMP\_IsCalibrationOutputSet**

|                                                   |                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_OPAMP_IsCalibrationOutputSet (OPAMP_TypeDef * OPAMPx)</code></b>                                                        |
| Function description                              | Get OPAMP calibration result of toggling output.                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• This functions returns: 0 if OPAMP calibration output is reset<br/>1 if OPAMP calibration output is set</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR OUTCAL LL_OPAMP_IsCalibrationOutputSet</li> </ul>                                                              |

### **LL\_OPAMP\_SetTrimmingValue**

|                                                   |                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_OPAMP_SetTrimmingValue (OPAMP_TypeDef * OPAMPx, uint32_t TransistorsDiffPair, uint32_t TrimmingValue)</code></b>                                                                                                                                                                                       |
| Function description                              | Set OPAMP trimming factor for the selected transistors differential pair NMOS or PMOS, corresponding to the selected power mode.                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> <li>• <b>TransistorsDiffPair:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_OPAMP_TRIMMING_NMOS</li> <li>– LL_OPAMP_TRIMMING_PMOS</li> </ul> </li> <li>• <b>TrimmingValue:</b> 0x00...0x1F</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR TRIMOFFSETN LL_OPAMP_SetTrimmingValue</li> <li>• CSR TRIMOFFSETP LL_OPAMP_SetTrimmingValue</li> </ul>                                                                                                                                                                                     |

### **LL\_OPAMP\_GetTrimmingValue**

|                      |                                                                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_OPAMP_GetTrimmingValue (OPAMP_TypeDef * OPAMPx, uint32_t TransistorsDiffPair)</code></b>                                                                                                                                                              |
| Function description | Get OPAMP trimming factor for the selected transistors differential pair NMOS or PMOS, corresponding to the selected power mode.                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>OPAMPx:</b> OPAMP instance</li> <li>• <b>TransistorsDiffPair:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_OPAMP_TRIMMING_NMOS</li> <li>– LL_OPAMP_TRIMMING_PMOS</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>0x0...0x1F:</b></li> </ul>                                                                                                                                                                                                                    |
| Reference Manual to  | <ul style="list-style-type: none"> <li>• CSR TRIMOFFSETN LL_OPAMP_GetTrimmingValue</li> </ul>                                                                                                                                                                                             |

|                         |                                                                                             |
|-------------------------|---------------------------------------------------------------------------------------------|
| LL API cross reference: | <ul style="list-style-type: none"> <li>CSR TRIMOFFSETP LL_OPAMP_GetTrimmingValue</li> </ul> |
|-------------------------|---------------------------------------------------------------------------------------------|

### LL\_OPAMP\_Enable

|                                             |                                                                                                                                                                                              |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_OPAMP_Enable (OPAMP_TypeDef * OPAMPx)</code></b>                                                                                                             |
| Function description                        | Enable OPAMP instance.                                                                                                                                                                       |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                              |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                |
| Notes                                       | <ul style="list-style-type: none"> <li>After enable from off state, OPAMP requires a delay to fulfill wake up time specification. Refer to device datasheet, parameter "tWAKEUP".</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CSR OPAMPXEN LL_OPAMP_Enable</li> </ul>                                                                                                               |

### LL\_OPAMP\_Disable

|                                             |                                                                                   |
|---------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_OPAMP_Disable (OPAMP_TypeDef * OPAMPx)</code></b> |
| Function description                        | Disable OPAMP instance.                                                           |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>OPAMPx:</b> OPAMP instance</li> </ul>   |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                     |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CSR OPAMPXEN LL_OPAMP_Disable</li> </ul>   |

### LL\_OPAMP\_IsEnabled

|                                             |                                                                                         |
|---------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_OPAMP_IsEnabled (OPAMP_TypeDef * OPAMPx)</code></b> |
| Function description                        | Get OPAMP instance enable state (0: OPAMP is disabled, 1: OPAMP is enabled)             |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>OPAMPx:</b> OPAMP instance</li> </ul>         |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>        |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CSR OPAMPXEN LL_OPAMP_IsEnabled</li> </ul>       |

### LL\_OPAMP\_Lock

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_OPAMP_Lock (OPAMP_TypeDef * OPAMPx)</code></b>  |
| Function description | Lock OPAMP instance.                                                            |
| Parameters           | <ul style="list-style-type: none"> <li><b>OPAMPx:</b> OPAMP instance</li> </ul> |

|                                                   |                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>Once locked, OPAMP configuration can be accessed in read-only.</li> <li>The only way to unlock the OPAMP is a device hardware reset.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CSR LOCK LL_OPAMP_Lock</li> </ul>                                                                                                               |

### LL\_OPAMP\_IsLocked

|                                                   |                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_OPAMP_IsLocked(<br/>OPAMP_TypeDef * OPAMPx)</code>                                                                                                   |
| Function description                              | Get OPAMP lock state (0: OPAMP is unlocked, 1: OPAMP is locked).                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>Once locked, OPAMP configuration can be accessed in read-only.</li> <li>The only way to unlock the OPAMP is a device hardware reset.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CSR LOCK LL_OPAMP_IsLocked</li> </ul>                                                                                                           |

### LL\_OPAMP\_DeInit

|                      |                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_OPAMP_DeInit (OPAMP_TypeDef * OPAMPx)</b>                                                                                                                                                                                                 |
| Function description | De-initialize registers of the selected OPAMP instance to their default reset values.                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li><b>OPAMPx:</b> OPAMP instance</li> </ul>                                                                                                                                                                             |
| Return values        | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>SUCCESS: OPAMP registers are de-initialized</li> <li>ERROR: OPAMP registers are not de-initialized</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>If comparator is locked, de-initialization by software is not possible. The only way to unlock the comparator is a device hardware reset.</li> </ul>                                                                 |

### LL\_OPAMP\_Init

|                      |                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_OPAMP_Init (OPAMP_TypeDef * OPAMPx,<br/>LL_OPAMP_InitTypeDef * OPAMP_InitStruct)</b>                                                        |
| Function description | Initialize some features of OPAMP instance.                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li><b>OPAMPx:</b> OPAMP instance</li> <li><b>OPAMP_InitStruct:</b> Pointer to a LL_OPAMP_InitTypeDef structure</li> </ul> |

|               |                                                                                                                                                                                                                                                       |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>SUCCESS: OPAMP registers are initialized</li> <li>ERROR: OPAMP registers are not initialized</li> </ul> </li> </ul> |
| Notes         | <ul style="list-style-type: none"> <li>This function reset bit of calibration mode to ensure to be in functional mode, in order to have OPAMP parameters (inputs selection, ...) set with the corresponding OPAMP mode to be effective.</li> </ul>    |

### LL\_OPAMP\_StructInit

|                      |                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_OPAMP_StructInit (LL_OPAMP_InitTypeDef * OPAMP_InitStruct)</b>                                                                                          |
| Function description | Set each LL_OPAMP_InitTypeDef field to default value.                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>OPAMP_InitStruct:</b> pointer to a LL_OPAMP_InitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                      |

## 72.3 OPAMP Firmware driver defines

### 72.3.1 OPAMP

#### *OPAMP functional mode*

|                                |                                                                                           |
|--------------------------------|-------------------------------------------------------------------------------------------|
| LL_OPAMP_MODE_STANDALONE       | OPAMP functional mode, OPAMP operation in standalone                                      |
| LL_OPAMP_MODE_FOLLOWER         | OPAMP functional mode, OPAMP operation in follower                                        |
| LL_OPAMP_MODE_PGA              | OPAMP functional mode, OPAMP operation in PGA                                             |
| LL_OPAMP_MODE_PGA_EXT_FILT_IO0 | OPAMP functional mode, OPAMP operation in PGA with external filtering on OPAMP input IO0. |
| LL_OPAMP_MODE_PGA_EXT_FILT_IO1 | OPAMP functional mode, OPAMP operation in PGA with external filtering on OPAMP input IO1. |

#### *Definitions of OPAMP hardware constraints delays*

LL\_OPAMP\_DELAY\_STARTUP\_US    Delay for OPAMP startup time

#### *OPAMP input inverting*

|                           |                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_OPAMP_INPUT_INVERT_IO0 | OPAMP inverting input connected to GPIO pin (pin PC5 for OPAMP1, pin PC5 for OPAMP2, pin PB10 for OPAMP3, pin PB10 for OPAMP4). Note: OPAMP inverting input is used with OPAMP in mode standalone or PGA with external capacitors for filtering circuit. Otherwise (OPAMP in mode follower), OPAMP inverting input is not used (not connected to GPIO pin). |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                  |                                                                                                                                                                                                                                                                                                                                                           |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_OPAMP_INPUT_INVERT_IO1        | OPAMP inverting input connected to GPIO pin (pin PA3 for OPAMP1, pin PA5 for OPAMP2, pin PB2 for OPAMP3, pin PD8 for OPAMP4). Note: OPAMP inverting input is used with OPAMP in mode standalone or PGA with external capacitors for filtering circuit. Otherwise (OPAMP in mode follower), OPAMP inverting input is not used (not connected to GPIO pin). |
| LL_OPAMP_INPUT_INVERT_CONNECT_NO | OPAMP inverting input not externally connected (intended for OPAMP in mode follower or PGA without external capacitors for filtering). Note: On this STM32 serie, this literal include cases of value 0x11 for mode follower and value 0x10 for mode PGA.                                                                                                 |

***OPAMP input inverting secondary***

|                               |                                                                                                                                                                                                                                                                                                                                                                       |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_OPAMP_INPUT_INVERT_IO0_SEC | OPAMP inverting input secondary connected to GPIO pin (pin PC5 for OPAMP1, pin PC5 for OPAMP2, pin PB10 for OPAMP3, pin PB10 for OPAMP4). Note: OPAMP inverting input is used with OPAMP in mode standalone or PGA with external capacitors for filtering circuit. Otherwise (OPAMP in mode follower), OPAMP inverting input is not used (not connected to GPIO pin). |
| LL_OPAMP_INPUT_INVERT_IO1_SEC | OPAMP inverting input secondary connected to GPIO pin (pin PA3 for OPAMP1, pin PA5 for OPAMP2, pin PB2 for OPAMP3, pin PD8 for OPAMP4). Note: OPAMP inverting input is used with OPAMP in mode standalone or PGA with external capacitors for filtering circuit. Otherwise (OPAMP in mode follower), OPAMP inverting input is not used (not connected to GPIO pin).   |

***OPAMP inputs multiplexer mode***

|                             |                                                                |
|-----------------------------|----------------------------------------------------------------|
| LL_OPAMP_INPUT_MUX_DISABLE  | OPAMP inputs multiplexer mode dosabled.                        |
| LL_OPAMP_INPUT_MUX_TIM1_CH6 | OPAMP inputs multiplexer mode enabled, controlled by TIM1 CC6. |

***OPAMP input non-inverting***

|                              |                                                                                                                                     |
|------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| LL_OPAMP_INPUT_NONINVERT_IO0 | OPAMP non inverting input connected to GPIO pin (pin PA1 for OPAMP1, pin PA7 for OPAMP2, pin PB0 for OPAMP3, pin PB13 for OPAMP4)   |
| LL_OPAMP_INPUT_NONINVERT_IO1 | OPAMP non inverting input connected to GPIO pin (pin PA7 for OPAMP1, pin PD14 for OPAMP2, pin PB13 for OPAMP3, pin PD11 for OPAMP4) |
| LL_OPAMP_INPUT_NONINVERT_IO2 | OPAMP non inverting input connected to GPIO pin (pin PA3 for OPAMP1, pin PB0 for OPAMP2, pin PA1 for OPAMP3, pin PB11 for           |

|                                                   |                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | OPAMP4)                                                                                                                                       |
| LL_OPAMP_INPUT_NONINVERT_IO3                      | OPAMP non inverting input connected to GPIO pin (pin PA5 for OPAMP1, pin PB14 for OPAMP2, pin PA5 for OPAMP3, pin PA4 for OPAMP4)             |
| LL_OPAMP_INPUT_NONINV_DAC1_CH1                    | OPAMP non inverting input connected to DAC1 channel1 output (specific to OPAMP instances: OPAMP4)                                             |
| LL_OPAMP_INPUT_NONINV_DAC1_CH2                    | OPAMP non inverting input connected to DAC1 channel2 output (specific to OPAMP instances: OPAMP1, OPAMP3)                                     |
| <b><i>OPAMP input non-inverting secondary</i></b> |                                                                                                                                               |
| LL_OPAMP_INPUT_NONINVERT_IO0_SEC                  | OPAMP non inverting input secondary connected to GPIO pin (pin PA1 for OPAMP1, pin PA7 for OPAMP2, pin PB0 for OPAMP3, pin PB13 for OPAMP4)   |
| LL_OPAMP_INPUT_NONINVERT_IO1_SEC                  | OPAMP non inverting input secondary connected to GPIO pin (pin PA7 for OPAMP1, pin PD14 for OPAMP2, pin PB13 for OPAMP3, pin PD11 for OPAMP4) |
| LL_OPAMP_INPUT_NONINVERT_IO2_SEC                  | OPAMP non inverting input secondary connected to GPIO pin (pin PA3 for OPAMP1, pin PB0 for OPAMP2, pin PA1 for OPAMP3, pin PB11 for OPAMP4)   |
| LL_OPAMP_INPUT_NONINVERT_IO3_SEC                  | OPAMP non inverting input secondary connected to GPIO pin (pin PA5 for OPAMP1, pin PD14 for OPAMP2, pin PA5 for OPAMP3, pin PA4 for OPAMP4)   |
| LL_OPAMP_INPUT_NONINV_DAC1_CH1_SEC                | OPAMP non inverting input secondary connected to DAC1 channel1 output (specific to OPAMP instances: OPAMP4)                                   |
| LL_OPAMP_INPUT_NONINV_DAC1_CH2_SEC                | OPAMP non inverting input secondary connected to DAC1 channel2 output (specific to OPAMP instances: OPAMP1, OPAMP3)                           |

***OPAMP mode calibration or functional.***

LL\_OPAMP\_MODE\_FUNCTIONAL OPAMP functional mode

LL\_OPAMP\_MODE\_CALIBRATION OPAMP calibration mode

***OPAMP PGA gain (relevant when OPAMP is in functional mode PGA)***

LL\_OPAMP\_PGA\_GAIN\_2 OPAMP PGA gain 2

LL\_OPAMP\_PGA\_GAIN\_4 OPAMP PGA gain 4

`LL_OPAMP_PGA_GAIN_8` OPAMP PGA gain 8

`LL_OPAMP_PGA_GAIN_16` OPAMP PGA gain 16

***OPAMP trimming mode***

`LL_OPAMP_TRIMMING_FACTORY` OPAMP trimming factors set to factory values

`LL_OPAMP_TRIMMING_USER` OPAMP trimming factors set to user values

***OPAMP trimming of transistors differential pair NMOS or PMOS***

`LL_OPAMP_TRIMMING_NMOS_VREF_90PC_VDDA` OPAMP trimming of transistors differential pair NMOS (internal reference voltage set to  $0.9^*Vdda$ ). Default parameters to be used for calibration using two trimming steps (one with each transistors differential pair NMOS and PMOS).

`LL_OPAMP_TRIMMING_NMOS_VREF_50PC_VDDA` OPAMP trimming of transistors differential pair NMOS (internal reference voltage set to  $0.5^*Vdda$ ).

`LL_OPAMP_TRIMMING_PMOS_VREF_10PC_VDDA` OPAMP trimming of transistors differential pair PMOS (internal reference voltage set to  $0.1^*Vdda$ ). Default parameters to be used for calibration using two trimming steps (one with each transistors differential pair NMOS and PMOS).

`LL_OPAMP_TRIMMING_PMOS_VREF_3_3PC_VDDA` OPAMP trimming of transistors differential pair PMOS (internal reference voltage set to  $0.33^*Vdda$ ).

`LL_OPAMP_TRIMMING_NMOS` OPAMP trimming of transistors differential pair NMOS (internal reference voltage set to  $0.9^*Vdda$ ). Default parameters to be used for calibration using two trimming steps (one with each transistors differential pair NMOS and PMOS).

`LL_OPAMP_TRIMMING_PMOS` OPAMP trimming of transistors differential pair PMOS (internal reference voltage set to  $0.1^*Vdda$ ). Default parameters to be used for calibration using two trimming steps (one with each transistors differential pair NMOS and PMOS).

***OPAMP internal reference voltage path state to output***

`LL_OPAMP_VREF_OUTPUT_DISABLE` OPAMP internal reference voltage path to

output is disabled.

`LL_OPAMP_VREF_OUTPUT_ENABLE` OPAMP internal reference voltage path to output is enabled.

***Common write and read registers macro***

`LL_OPAMP_WriteReg` **Description:**

- Write a value in OPAMP register.

**Parameters:**

- `__INSTANCE__`: OPAMP Instance
- `__REG__`: Register to be written
- `__VALUE__`: Value to be written in the register

**Return value:**

- None

`LL_OPAMP_ReadReg` **Description:**

- Read a value in OPAMP register.

**Parameters:**

- `__INSTANCE__`: OPAMP Instance
- `__REG__`: Register to be read

**Return value:**

- Register: value

## 73 LL PWR Generic Driver

### 73.1 PWR Firmware driver API description

#### 73.1.1 Detailed description of functions

##### **LL\_PWR\_EnableSDADC**

|                                             |                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_PWR_EnableSDADC (uint32_t Analogx)</code>                                                                                                                                                                                                                                                     |
| Function description                        | Enables the SDADC peripheral functionality.                                                                                                                                                                                                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>Analogx:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>- <code>LL_PWR_SDADC_ANALOG1</code></li> <li>- <code>LL_PWR_SDADC_ANALOG2</code></li> <li>- <code>LL_PWR_SDADC_ANALOG3</code></li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                             |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR ENSD1 <code>LL_PWR_EnableSDADC</code></li> <li>• CR ENSD2 <code>LL_PWR_EnableSDADC</code></li> <li>• CR ENSD3 <code>LL_PWR_EnableSDADC</code></li> </ul>                                                                                                                        |

##### **LL\_PWR\_DisableSDADC**

|                                             |                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_PWR_DisableSDADC (uint32_t Analogx)</code>                                                                                                                                                                                                                                                    |
| Function description                        | Disables the SDADC peripheral functionality.                                                                                                                                                                                                                                                                                |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>Analogx:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>- <code>LL_PWR_SDADC_ANALOG1</code></li> <li>- <code>LL_PWR_SDADC_ANALOG2</code></li> <li>- <code>LL_PWR_SDADC_ANALOG3</code></li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                             |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR ENSD1 <code>LL_PWR_EnableSDADC</code></li> <li>• CR ENSD2 <code>LL_PWR_EnableSDADC</code></li> <li>• CR ENSD3 <code>LL_PWR_EnableSDADC</code></li> </ul>                                                                                                                        |

##### **LL\_PWR\_IsEnabledSDADC**

|                      |                                                                                                                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_PWR_IsEnabledSDADC (uint32_t Analogx)</code>                                                                                                                                                                                                 |
| Function description | Check if SDADCx has been enabled or not.                                                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Analogx:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>- <code>LL_PWR_SDADC_ANALOG1</code></li> <li>- <code>LL_PWR_SDADC_ANALOG2</code></li> </ul> </li> </ul> |

– LL\_PWR\_SDADC\_ANALOG3

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CR ENSD1 LL\_PWR\_IsEnabledSDADC
- CR ENSD2 LL\_PWR\_IsEnabledSDADC
- CR ENSD3 LL\_PWR\_IsEnabledSDADC

### **LL\_PWR\_EnableBkUpAccess**

Function name **\_\_STATIC\_INLINE void LL\_PWR\_EnableBkUpAccess (void )**

Function description Enable access to the backup domain.

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CR DBP LL\_PWR\_EnableBkUpAccess

### **LL\_PWR\_DisableBkUpAccess**

Function name **\_\_STATIC\_INLINE void LL\_PWR\_DisableBkUpAccess (void )**

Function description Disable access to the backup domain.

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CR DBP LL\_PWR\_DisableBkUpAccess

### **LL\_PWR\_IsEnabledBkUpAccess**

Function name **\_\_STATIC\_INLINE uint32\_t LL\_PWR\_IsEnabledBkUpAccess (void )**

Function description Check if the backup domain is enabled.

Return values

- **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- CR DBP LL\_PWR\_IsEnabledBkUpAccess

### **LL\_PWR\_SetRegulModeDS**

Function name **\_\_STATIC\_INLINE void LL\_PWR\_SetRegulModeDS (uint32\_t RegulMode)**

Function description Set voltage Regulator mode during deep sleep mode.

Parameters

- **RegulMode:** This parameter can be one of the following values:
  - LL\_PWR\_REGU\_DSMODE\_MAIN
  - LL\_PWR\_REGU\_DSMODE\_LOW\_POWER

Return values

- **None**

Reference Manual to  
LL API cross

- CR LPDS LL\_PWR\_SetRegulModeDS

reference:

### **LL\_PWR\_GetRegulModeDS**

|                                                   |                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_PWR_GetRegulModeDS (void )</code></b>                                                                                                                                                                                               |
| Function description                              | Get voltage Regulator mode during deep sleep mode.                                                                                                                                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- <code>LL_PWR_REGU_DSMODE_MAIN</code></li> <li>- <code>LL_PWR_REGU_DSMODE_LOW_POWER</code></li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR LPDS LL_PWR_GetRegulModeDS</li> </ul>                                                                                                                                                                                       |

### **LL\_PWR\_SetPowerMode**

|                                                   |                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_PWR_SetPowerMode (uint32_t PDMode)</code></b>                                                                                                                                                                                                                                           |
| Function description                              | Set Power Down mode when CPU enters deepsleep.                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>PDMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- <code>LL_PWR_MODE_STOP_MAINREGU</code></li> <li>- <code>LL_PWR_MODE_STOP_LPREGU</code></li> <li>- <code>LL_PWR_MODE_STANDBY</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR PDDS LL_PWR_SetPowerMode</li> <li>• CR LPDS LL_PWR_SetPowerMode</li> </ul>                                                                                                                                                                                                  |

### **LL\_PWR\_GetPowerMode**

|                                                   |                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_PWR_GetPowerMode (void )</code></b>                                                                                                                                                                                                                                          |
| Function description                              | Get Power Down mode when CPU enters deepsleep.                                                                                                                                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- <code>LL_PWR_MODE_STOP_MAINREGU</code></li> <li>- <code>LL_PWR_MODE_STOP_LPREGU</code></li> <li>- <code>LL_PWR_MODE_STANDBY</code></li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR PDDS LL_PWR_GetPowerMode</li> <li>• CR LPDS LL_PWR_GetPowerMode</li> </ul>                                                                                                                                                                                           |

### **LL\_PWR\_SetPVDLevel**

|                      |                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_PWR_SetPVDLevel (uint32_t PVDLevel)</code></b>                                          |
| Function description | Configure the voltage threshold detected by the Power Voltage Detector.                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>PVDLevel:</b> This parameter can be one of the following values:</li> </ul> |

- LL\_PWR\_PVDLEVEL\_0
- LL\_PWR\_PVDLEVEL\_1
- LL\_PWR\_PVDLEVEL\_2
- LL\_PWR\_PVDLEVEL\_3
- LL\_PWR\_PVDLEVEL\_4
- LL\_PWR\_PVDLEVEL\_5
- LL\_PWR\_PVDLEVEL\_6
- LL\_PWR\_PVDLEVEL\_7

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CR PLS LL\_PWR\_SetPVDLevel

### **LL\_PWR\_GetPVDLevel**

Function name **\_\_STATIC\_INLINE uint32\_t LL\_PWR\_GetPVDLevel (void )**

Function description Get the voltage threshold detection.

Return values

- **Returned:** value can be one of the following values:

- LL\_PWR\_PVDLEVEL\_0
- LL\_PWR\_PVDLEVEL\_1
- LL\_PWR\_PVDLEVEL\_2
- LL\_PWR\_PVDLEVEL\_3
- LL\_PWR\_PVDLEVEL\_4
- LL\_PWR\_PVDLEVEL\_5
- LL\_PWR\_PVDLEVEL\_6
- LL\_PWR\_PVDLEVEL\_7

Reference Manual to  
LL API cross  
reference:

- CR PLS LL\_PWR\_GetPVDLevel

### **LL\_PWR\_EnablePVD**

Function name **\_\_STATIC\_INLINE void LL\_PWR\_EnablePVD (void )**

Function description Enable Power Voltage Detector.

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CR PVDE LL\_PWR\_EnablePVD

### **LL\_PWR\_DisablePVD**

Function name **\_\_STATIC\_INLINE void LL\_PWR\_DisablePVD (void )**

Function description Disable Power Voltage Detector.

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CR PVDE LL\_PWR\_DisablePVD

**LL\_PWR\_IsEnabledPVD**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_PWR_IsEnabledPVD (void )</code></b>            |
| Function description                              | Check if Power Voltage Detector is enabled.                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR PVDE LL_PWR_IsEnabledPVD</li> </ul>    |

**LL\_PWR\_EnableWakeUpPin**

|                                                   |                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_PWR_EnableWakeUpPin (uint32_t WakeUpPin)</code></b>                                                                                                                                                                                                               |
| Function description                              | Enable the WakeUp PINx functionality.                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>WakeUpPin:</b> This parameter can be one of the following values: (*) not available on all devices <ul style="list-style-type: none"> <li>- LL_PWR_WAKEUP_PIN1</li> <li>- LL_PWR_WAKEUP_PIN2</li> <li>- LL_PWR_WAKEUP_PIN3 (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR EWUP1 LL_PWR_EnableWakeUpPin</li> <li>• CSR EWUP2 LL_PWR_EnableWakeUpPin</li> <li>• CSR EWUP3 LL_PWR_EnableWakeUpPin</li> </ul>                                                                                                                      |

**LL\_PWR\_DisableWakeUpPin**

|                                                   |                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_PWR_DisableWakeUpPin (uint32_t WakeUpPin)</code></b>                                                                                                                                                                                                              |
| Function description                              | Disable the WakeUp PINx functionality.                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>WakeUpPin:</b> This parameter can be one of the following values: (*) not available on all devices <ul style="list-style-type: none"> <li>- LL_PWR_WAKEUP_PIN1</li> <li>- LL_PWR_WAKEUP_PIN2</li> <li>- LL_PWR_WAKEUP_PIN3 (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CSR EWUP1 LL_PWR_DisableWakeUpPin</li> <li>• CSR EWUP2 LL_PWR_DisableWakeUpPin</li> <li>• CSR EWUP3 LL_PWR_DisableWakeUpPin</li> </ul>                                                                                                                   |

**LL\_PWR\_IsEnabledWakeUpPin**

Function name      **`_STATIC_INLINE uint32_t LL_PWR_IsEnabledWakeUpPin  
(uint32_t WakeUpPin)`**

Function description      Check if the WakeUp PINx functionality is enabled.

Parameters     
 

- **WakeUpPin:** This parameter can be one of the following values: (\*) not available on all devices
  - `LL_PWR_WAKEUP_PIN1`
  - `LL_PWR_WAKEUP_PIN2`
  - `LL_PWR_WAKEUP_PIN3` (\*)

Return values     
 

- **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:     
 

- CSR EWUP1 LL\_PWR\_IsEnabledWakeUpPin
- CSR EWUP2 LL\_PWR\_IsEnabledWakeUpPin
- CSR EWUP3 LL\_PWR\_IsEnabledWakeUpPin

**LL\_PWR\_IsActiveFlag\_WU**

Function name      **`_STATIC_INLINE uint32_t LL_PWR_IsActiveFlag_WU (void )`**

Function description      Get Wake-up Flag.

Return values     
 

- **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:     
 

- CSR WUF LL\_PWR\_IsActiveFlag\_WU

**LL\_PWR\_IsActiveFlag\_SB**

Function name      **`_STATIC_INLINE uint32_t LL_PWR_IsActiveFlag_SB (void )`**

Function description      Get Standby Flag.

Return values     
 

- **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:     
 

- CSR SBF LL\_PWR\_IsActiveFlag\_SB

**LL\_PWR\_IsActiveFlag\_PVDO**

Function name      **`_STATIC_INLINE uint32_t LL_PWR_IsActiveFlag_PVDO (void )`**

Function description      Indicate whether VDD voltage is below the selected PVD threshold.

Return values     
 

- **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:     
 

- CSR PVDO LL\_PWR\_IsActiveFlag\_PVDO

**LL\_PWR\_IsActiveFlag\_VREFINTRDY**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_PWR_IsActiveFlag_VREFINTRDY (void )</code>   |
| Function description                              | Get Internal Reference VrefInt Flag.                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | • CSR VREFINTRDYF LL_PWR_IsActiveFlag_VREFINTRDY                                   |

**LL\_PWR\_ClearFlag\_SB**

|                                                   |                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_PWR_ClearFlag_SB (void )</code>   |
| Function description                              | Clear Standby Flag.                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | • CR CSBF LL_PWR_ClearFlag_SB                                   |

**LL\_PWR\_ClearFlag\_WU**

|                                                   |                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_PWR_ClearFlag_WU (void )</code>   |
| Function description                              | Clear Wake-up Flags.                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | • CR CWUF LL_PWR_ClearFlag_WU                                   |

**LL\_PWR\_DelInit**

|                      |                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_PWR_DelInit (void )</code>                                                                                                                                                                                         |
| Function description | De-initialize the PWR registers to their default reset values.                                                                                                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: PWR registers are de-initialized</li> <li>– ERROR: not applicable</li> </ul> </li> </ul> |

## 73.2 PWR Firmware driver defines

### 73.2.1 PWR

***Clear Flags Defines***

|                             |                    |
|-----------------------------|--------------------|
| <code>LL_PWR_CR_CSBF</code> | Clear standby flag |
| <code>LL_PWR_CR_CWUF</code> | Clear wakeup flag  |

***Get Flags Defines***

|                             |              |
|-----------------------------|--------------|
| <code>LL_PWR_CSR_WUF</code> | Wakeup flag  |
| <code>LL_PWR_CSR_SBF</code> | Standby flag |

---

|                        |                                    |
|------------------------|------------------------------------|
| LL_PWR_CSR_PVDO        | Power voltage detector output flag |
| LL_PWR_CSR_VREFINTRDYF | VREFINT ready flag                 |
| LL_PWR_CSR_EWUP1       | Enable WKUP pin 1                  |
| LL_PWR_CSR_EWUP2       | Enable WKUP pin 2                  |
| LL_PWR_CSR_EWUP3       | Enable WKUP pin 3                  |

**Mode Power**

|                           |                                                                             |
|---------------------------|-----------------------------------------------------------------------------|
| LL_PWR_MODE_STOP_MAINREGU | Enter Stop mode when the CPU enters deepsleep                               |
| LL_PWR_MODE_STOP_LPREGU   | Enter Stop mode (with low power Regulator ON) when the CPU enters deepsleep |
| LL_PWR_MODE_STANDBY       | Enter Standby mode when the CPU enters deepsleep                            |

**Power Voltage Detector Level**

|                   |                                         |
|-------------------|-----------------------------------------|
| LL_PWR_PVDLEVEL_0 | Voltage threshold detected by PVD 2.2 V |
| LL_PWR_PVDLEVEL_1 | Voltage threshold detected by PVD 2.3 V |
| LL_PWR_PVDLEVEL_2 | Voltage threshold detected by PVD 2.4 V |
| LL_PWR_PVDLEVEL_3 | Voltage threshold detected by PVD 2.5 V |
| LL_PWR_PVDLEVEL_4 | Voltage threshold detected by PVD 2.6 V |
| LL_PWR_PVDLEVEL_5 | Voltage threshold detected by PVD 2.7 V |
| LL_PWR_PVDLEVEL_6 | Voltage threshold detected by PVD 2.8 V |
| LL_PWR_PVDLEVEL_7 | Voltage threshold detected by PVD 2.9 V |

**Regulator Mode In Deep Sleep Mode**

|                              |                                                           |
|------------------------------|-----------------------------------------------------------|
| LL_PWR_REGU_DSMODE_MAIN      | Voltage Regulator in main mode during deepsleep mode      |
| LL_PWR_REGU_DSMODE_LOW_POWER | Voltage Regulator in low-power mode during deepsleep mode |

**Wakeup Pins**

|                    |                                             |
|--------------------|---------------------------------------------|
| LL_PWR_WAKEUP_PIN1 | WKUP pin 1 : PA0                            |
| LL_PWR_WAKEUP_PIN2 | WKUP pin 2 : PC13                           |
| LL_PWR_WAKEUP_PIN3 | WKUP pin 3 : PE6 or PA2 according to device |

**Common write and read registers Macros**

|                 |                                                                                    |
|-----------------|------------------------------------------------------------------------------------|
| LL_PWR_WriteReg | <b>Description:</b>                                                                |
|                 | <ul style="list-style-type: none"> <li>• Write a value in PWR register.</li> </ul> |

**Parameters:**

- \_\_REG\_\_: Register to be written
- \_\_VALUE\_\_: Value to be written in the register

**Return value:**

- None

[LL\\_PWR\\_ReadReg](#)**Description:**

- Read a value in PWR register.

**Parameters:**

- \_\_REG\_\_: Register to be read

**Return value:**

- Register: value

## 74 LL RCC Generic Driver

### 74.1 RCC Firmware driver registers structures

#### 74.1.1 LL\_RCC\_ClocksTypeDef

##### Data Fields

- *uint32\_t SYSCLK\_Frequency*
- *uint32\_t HCLK\_Frequency*
- *uint32\_t PCLK1\_Frequency*
- *uint32\_t PCLK2\_Frequency*

##### Field Documentation

- *uint32\_t LL\_RCC\_ClocksTypeDef::SYSCLK\_Frequency*  
SYSCLK clock frequency
- *uint32\_t LL\_RCC\_ClocksTypeDef::HCLK\_Frequency*  
HCLK clock frequency
- *uint32\_t LL\_RCC\_ClocksTypeDef::PCLK1\_Frequency*  
PCLK1 clock frequency
- *uint32\_t LL\_RCC\_ClocksTypeDef::PCLK2\_Frequency*  
PCLK2 clock frequency

### 74.2 RCC Firmware driver API description

#### 74.2.1 Detailed description of functions

##### LL\_RCC\_HSE\_EnableCSS

|                                                   |                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_HSE_EnableCSS (void )</code>                  |
| Function description                              | Enable the Clock Security System.                                               |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR CSSON LL_RCC_HSE_EnableCSS</li></ul> |

##### LL\_RCC\_HSE\_DisableCSS

|                      |                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RCC_HSE_DisableCSS (void )</code>                                         |
| Function description | Disable the Clock Security System.                                                                      |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                           |
| Notes                | <ul style="list-style-type: none"><li>• Cannot be disabled in HSE is ready (only by hardware)</li></ul> |

**LL\_RCC\_HSE\_EnableBypass**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_HSE_EnableBypass (void )</code></b>            |
| Function description                              | Enable HSE external oscillator (HSE Bypass)                                         |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR HSEBYP LL_RCC_HSE_EnableBypass</li></ul> |

**LL\_RCC\_HSE\_DisableBypass**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_HSE_DisableBypass (void )</code></b>            |
| Function description                              | Disable HSE external oscillator (HSE Bypass)                                         |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR HSEBYP LL_RCC_HSE_DisableBypass</li></ul> |

**LL\_RCC\_HSE\_Enable**

|                                                   |                                                                              |
|---------------------------------------------------|------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_HSE_Enable (void )</code></b>           |
| Function description                              | Enable HSE crystal oscillator (HSE ON)                                       |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR HSEON LL_RCC_HSE_Enable</li></ul> |

**LL\_RCC\_HSE\_Disable**

|                                                   |                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_HSE_Disable (void )</code></b>           |
| Function description                              | Disable HSE crystal oscillator (HSE ON)                                       |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR HSEON LL_RCC_HSE_Disable</li></ul> |

**LL\_RCC\_HSE\_IsReady**

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RCC_HSE_IsReady (void )</code></b>          |
| Function description                              | Check if HSE oscillator Ready.                                                   |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR HSERDY LL_RCC_HSE_IsReady</li></ul>   |

**LL\_RCC\_HSI\_Enable**

|                                                   |                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_HSI_Enable (void )</code></b>             |
| Function description                              | Enable HSI oscillator.                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR HSION LL_RCC_HSI_Enable</li> </ul> |

**LL\_RCC\_HSI\_Disable**

|                                                   |                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_HSI_Disable (void )</code></b>             |
| Function description                              | Disable HSI oscillator.                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR HSION LL_RCC_HSI_Disable</li> </ul> |

**LL\_RCC\_HSI\_IsReady**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RCC_HSI_IsReady (void )</code></b>            |
| Function description                              | Check if HSI clock is ready.                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR HSIRDY LL_RCC_HSI_IsReady</li> </ul>   |

**LL\_RCC\_HSI\_GetCalibration**

|                                                   |                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RCC_HSI_GetCalibration (void )</code></b>                                                                    |
| Function description                              | Get HSI Calibration value.                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Between:</b> Min_Data = 0x00 and Max_Data = 0xFF</li> </ul>                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• When HSITRIM is written, HSICAL is updated with the sum of HSITRIM and the factory trim value</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR HSICAL LL_RCC_HSI_GetCalibration</li> </ul>                                                           |

**LL\_RCC\_HSI\_SetCalibTrimming**

|                      |                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_RCC_HSI_SetCalibTrimming (uint32_t Value)</code></b>                         |
| Function description | Set HSI Calibration trimming.                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data = 0x00 and Max_Data = 0x1F</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                               |

---

|                                                   |                                                                                                                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                             | <ul style="list-style-type: none"> <li>user-programmable trimming value that is added to the HSICAL</li> <li>Default value is 16, which, when added to the HSICAL value, should trim the HSI to 16 MHz +/- 1 %</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR HSITRIM LL_RCC_HSI_SetCalibTrimming</li> </ul>                                                                                                                                  |

### LL\_RCC\_HSI\_GetCalibTrimming

|                                                   |                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_HSI_GetCalibTrimming(void)</code>                               |
| Function description                              | Get HSI Calibration trimming.                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Between:</b> Min_Data = 0x00 and Max_Data = 0x1F</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR HSITRIM LL_RCC_HSI_SetCalibTrimming</li> </ul>              |

### LL\_RCC\_LSE\_Enable

|                                                   |                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_LSE_Enable(void)</code>                      |
| Function description                              | Enable Low Speed External (LSE) crystal.                                       |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>BDCR LSEON LL_RCC_LSE_Enable</li> </ul> |

### LL\_RCC\_LSE\_Disable

|                                                   |                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_LSE_Disable(void)</code>                      |
| Function description                              | Disable Low Speed External (LSE) crystal.                                       |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>BDCR LSEON LL_RCC_LSE_Disable</li> </ul> |

### LL\_RCC\_LSE\_EnableBypass

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_LSE_EnableBypass(void)</code>                       |
| Function description                              | Enable external clock source (LSE bypass).                                            |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>BDCR LSEBYP LL_RCC_LSE_EnableBypass</li> </ul> |

**LL\_RCC\_LSE\_DisableBypass**

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_RCC_LSE_DisableBypass (void )</code></b>                 |
| Function description                              | Disable external clock source (LSE bypass).                                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDCR LSEBYP LL_RCC_LSE_DisableBypass</li> </ul> |

**LL\_RCC\_LSE\_SetDriveCapability**

|                                                   |                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_RCC_LSE_SetDriveCapability (uint32_t LSEDrive)</code></b>                                                                                                                                                                                                                   |
| Function description                              | Set LSE oscillator drive capability.                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>LSEDrive:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_RCC_LSEDRIVE_LOW</li> <li>– LL_RCC_LSEDRIVE_MEDIUMLOW</li> <li>– LL_RCC_LSEDRIVE_MEDIUMHIGH</li> <li>– LL_RCC_LSEDRIVE_HIGH</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• The oscillator is in Xtal mode when it is not in bypass mode.</li> </ul>                                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDCR LSEDRV LL_RCC_LSE_SetDriveCapability</li> </ul>                                                                                                                                                                                                               |

**LL\_RCC\_LSE\_GetDriveCapability**

|                                                   |                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_RCC_LSE_GetDriveCapability (void )</code></b>                                                                                                                                                                                                                  |
| Function description                              | Get LSE oscillator drive capability.                                                                                                                                                                                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_RCC_LSEDRIVE_LOW</li> <li>– LL_RCC_LSEDRIVE_MEDIUMLOW</li> <li>– LL_RCC_LSEDRIVE_MEDIUMHIGH</li> <li>– LL_RCC_LSEDRIVE_HIGH</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDCR LSEDRV LL_RCC_LSE_GetDriveCapability</li> </ul>                                                                                                                                                                                                      |

**LL\_RCC\_LSE\_IsReady**

|                                     |                                                                                    |
|-------------------------------------|------------------------------------------------------------------------------------|
| Function name                       | <b><code>_STATIC_INLINE uint32_t LL_RCC_LSE_IsReady (void )</code></b>             |
| Function description                | Check if LSE oscillator Ready.                                                     |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• BDCR LSERDY LL_RCC_LSE_IsReady</li> </ul> |

reference:

### LL\_RCC\_LSI\_Enable

Function name **`__STATIC_INLINE void LL_RCC_LSI_Enable (void )`**

Function description Enable LSI Oscillator.

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
CSR LSION LL\_RCC\_LSI\_Enable

### LL\_RCC\_LSI\_Disable

Function name **`__STATIC_INLINE void LL_RCC_LSI_Disable (void )`**

Function description Disable LSI Oscillator.

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
CSR LSIRDY LL\_RCC\_LSI\_Disable

### LL\_RCC\_LSI\_IsReady

Function name **`__STATIC_INLINE uint32_t LL_RCC_LSI_IsReady (void )`**

Function description Check if LSI is Ready.

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
CSR LSIRDY LL\_RCC\_LSI\_IsReady

### LL\_RCC\_SetSysClkSource

Function name **`__STATIC_INLINE void LL_RCC_SetSysClkSource (uint32_t Source)`**

Function description Configure the system clock source.

Parameters • **Source:** This parameter can be one of the following values:

- `LL_RCC_SYS_CLKSOURCE_HSI`
- `LL_RCC_SYS_CLKSOURCE_HSE`
- `LL_RCC_SYS_CLKSOURCE_PLL`

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
CFGREG SW LL\_RCC\_SetSysClkSource

**LL\_RCC\_GetSysClkSource**

|                                                   |                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_GetSysClkSource (void )</code>                                                                                                                                                                                                                                |
| Function description                              | Get the system clock source.                                                                                                                                                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RCC_SYS_CLKSOURCE_STATUS_HSI</li> <li>– LL_RCC_SYS_CLKSOURCE_STATUS_HSE</li> <li>– LL_RCC_SYS_CLKSOURCE_STATUS_PLL</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR SWS LL_RCC_GetSysClkSource</li> </ul>                                                                                                                                                                                                                 |

**LL\_RCC\_SetAHBPrescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_SetAHBPrescaler (uint32_t Prescaler)</code>                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Set AHB prescaler.                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Prescaler:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RCC_SYSCLK_DIV_1</li> <li>– LL_RCC_SYSCLK_DIV_2</li> <li>– LL_RCC_SYSCLK_DIV_4</li> <li>– LL_RCC_SYSCLK_DIV_8</li> <li>– LL_RCC_SYSCLK_DIV_16</li> <li>– LL_RCC_SYSCLK_DIV_64</li> <li>– LL_RCC_SYSCLK_DIV_128</li> <li>– LL_RCC_SYSCLK_DIV_256</li> <li>– LL_RCC_SYSCLK_DIV_512</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR HPRE LL_RCC_SetAHBPrescaler</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                        |

**LL\_RCC\_SetAPB1Prescaler**

|                                     |                                                                                                                                                                                                                                                                                                                                |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                       | <code>__STATIC_INLINE void LL_RCC_SetAPB1Prescaler (uint32_t Prescaler)</code>                                                                                                                                                                                                                                                 |
| Function description                | Set APB1 prescaler.                                                                                                                                                                                                                                                                                                            |
| Parameters                          | <ul style="list-style-type: none"> <li>• <b>Prescaler:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RCC_APB1_DIV_1</li> <li>– LL_RCC_APB1_DIV_2</li> <li>– LL_RCC_APB1_DIV_4</li> <li>– LL_RCC_APB1_DIV_8</li> <li>– LL_RCC_APB1_DIV_16</li> </ul> </li> </ul> |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• CFGR PPRE1 LL_RCC_SetAPB1Prescaler</li> </ul>                                                                                                                                                                                                                                         |

reference:

### **LL\_RCC\_SetAPB2Prescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_SetAPB2Prescaler (uint32_t Prescaler)</code></b>                                                                                                                                                                                                                                                                                                           |
| Function description                              | Set APB2 prescaler.                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Prescaler:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– <code>LL_RCC_APB2_DIV_1</code></li> <li>– <code>LL_RCC_APB2_DIV_2</code></li> <li>– <code>LL_RCC_APB2_DIV_4</code></li> <li>– <code>LL_RCC_APB2_DIV_8</code></li> <li>– <code>LL_RCC_APB2_DIV_16</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR PPRE2 LL_RCC_SetAPB2Prescaler</li> </ul>                                                                                                                                                                                                                                                                                                          |

### **LL\_RCC\_GetAHBPrescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RCC_GetAHBPrescaler (void )</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Get AHB prescaler.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– <code>LL_RCC_SYSCLK_DIV_1</code></li> <li>– <code>LL_RCC_SYSCLK_DIV_2</code></li> <li>– <code>LL_RCC_SYSCLK_DIV_4</code></li> <li>– <code>LL_RCC_SYSCLK_DIV_8</code></li> <li>– <code>LL_RCC_SYSCLK_DIV_16</code></li> <li>– <code>LL_RCC_SYSCLK_DIV_64</code></li> <li>– <code>LL_RCC_SYSCLK_DIV_128</code></li> <li>– <code>LL_RCC_SYSCLK_DIV_256</code></li> <li>– <code>LL_RCC_SYSCLK_DIV_512</code></li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR HPRE LL_RCC_GetAHBPrescaler</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

### **LL\_RCC\_GetAPB1Prescaler**

|                      |                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_RCC_GetAPB1Prescaler (void )</code></b>                                                                                                                                                                                                                                                                                                          |
| Function description | Get APB1 prescaler.                                                                                                                                                                                                                                                                                                                                                                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– <code>LL_RCC_APB1_DIV_1</code></li> <li>– <code>LL_RCC_APB1_DIV_2</code></li> <li>– <code>LL_RCC_APB1_DIV_4</code></li> <li>– <code>LL_RCC_APB1_DIV_8</code></li> <li>– <code>LL_RCC_APB1_DIV_16</code></li> </ul> </li> </ul> |

- Reference Manual to  
LL API cross  
reference:
- CFGR\_PPREG1\_LL\_RCC\_GetAPB1Prescaler

### **LL\_RCC\_GetAPB2Prescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RCC_GetAPB2Prescaler (void )</code></b>                                                                                                                                                                                                                                         |
| Function description                              | Get APB2 prescaler.                                                                                                                                                                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RCC_APB2_DIV_1</li> <li>– LL_RCC_APB2_DIV_2</li> <li>– LL_RCC_APB2_DIV_4</li> <li>– LL_RCC_APB2_DIV_8</li> <li>– LL_RCC_APB2_DIV_16</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR_PPREG2_LL_RCC_GetAPB2Prescaler</li> </ul>                                                                                                                                                                                                                              |

### **LL\_RCC\_ConfigMCO**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_ConfigMCO (uint32_t MCOxSource, uint32_t MCOxPrescaler)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Configure MCOx.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>MCOxSource:</b> This parameter can be one of the following values: (*) value not defined in all devices           <ul style="list-style-type: none"> <li>– LL_RCC_MCO1SOURCE_NOCLK</li> <li>– LL_RCC_MCO1SOURCE_SYSCLK</li> <li>– LL_RCC_MCO1SOURCE_HSI</li> <li>– LL_RCC_MCO1SOURCE_HSE</li> <li>– LL_RCC_MCO1SOURCE_LSI</li> <li>– LL_RCC_MCO1SOURCE_LSE</li> <li>– LL_RCC_MCO1SOURCE_PLLCLK (*)</li> <li>– LL_RCC_MCO1SOURCE_PLLCLK_DIV_2</li> </ul> </li> <li>• <b>MCOxPrescaler:</b> This parameter can be one of the following values: (*) value not defined in all devices           <ul style="list-style-type: none"> <li>– LL_RCC_MCO1_DIV_1</li> <li>– LL_RCC_MCO1_DIV_2 (*)</li> <li>– LL_RCC_MCO1_DIV_4 (*)</li> <li>– LL_RCC_MCO1_DIV_8 (*)</li> <li>– LL_RCC_MCO1_DIV_16 (*)</li> <li>– LL_RCC_MCO1_DIV_32 (*)</li> <li>– LL_RCC_MCO1_DIV_64 (*)</li> <li>– LL_RCC_MCO1_DIV_128 (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR_MCO_LL_RCC_ConfigMCO</li> <li>• CFGR_MCOPRE_LL_RCC_ConfigMCO</li> <li>• CFGR_PLLNODIV_LL_RCC_ConfigMCO</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

**LL\_RCC\_SetUSARTClockSource**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_RCC_SetUSARTClockSource<br/>(uint32_t USARTxSource)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Configure USARTx clock source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTxSource:</b> This parameter can be one of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- <code>LL_RCC_USART1_CLKSOURCE_PCLK1</code> (*)</li> <li>- <code>LL_RCC_USART1_CLKSOURCE_PCLK2</code> (*)</li> <li>- <code>LL_RCC_USART1_CLKSOURCE_SYSCLK</code></li> <li>- <code>LL_RCC_USART1_CLKSOURCE_LSE</code></li> <li>- <code>LL_RCC_USART1_CLKSOURCE_HSI</code></li> <li>- <code>LL_RCC_USART2_CLKSOURCE_PCLK1</code> (*)</li> <li>- <code>LL_RCC_USART2_CLKSOURCE_SYSCLK</code> (*)</li> <li>- <code>LL_RCC_USART2_CLKSOURCE_LSE</code> (*)</li> <li>- <code>LL_RCC_USART2_CLKSOURCE_HSI</code> (*)</li> <li>- <code>LL_RCC_USART3_CLKSOURCE_PCLK1</code> (*)</li> <li>- <code>LL_RCC_USART3_CLKSOURCE_SYSCLK</code> (*)</li> <li>- <code>LL_RCC_USART3_CLKSOURCE_LSE</code> (*)</li> <li>- <code>LL_RCC_USART3_CLKSOURCE_HSI</code> (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>CFG3 USART1SW LL_RCC_SetUSARTClockSource</code></li> <li>• <code>CFG3 USART2SW LL_RCC_SetUSARTClockSource</code></li> <li>• <code>CFG3 USART3SW LL_RCC_SetUSARTClockSource</code></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

**LL\_RCC\_SetUARTClockSource**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_RCC_SetUARTClockSource<br/>(uint32_t UARTxSource)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Configure UARTx clock source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>UARTxSource:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_RCC_UART4_CLKSOURCE_PCLK1</code></li> <li>- <code>LL_RCC_UART4_CLKSOURCE_SYSCLK</code></li> <li>- <code>LL_RCC_UART4_CLKSOURCE_LSE</code></li> <li>- <code>LL_RCC_UART4_CLKSOURCE_HSI</code></li> <li>- <code>LL_RCC_UART5_CLKSOURCE_PCLK1</code></li> <li>- <code>LL_RCC_UART5_CLKSOURCE_SYSCLK</code></li> <li>- <code>LL_RCC_UART5_CLKSOURCE_LSE</code></li> <li>- <code>LL_RCC_UART5_CLKSOURCE_HSI</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>CFG3 UART4SW LL_RCC_SetUARTClockSource</code></li> <li>• <code>CFG3 UART5SW LL_RCC_SetUARTClockSource</code></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                 |

**LL\_RCC\_SetI2CClockSource**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_SetI2CClockSource (uint32_t I2CxSource)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Configure I2Cx clock source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2CxSource:</b> This parameter can be one of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- <code>LL_RCC_I2C1_CLKSOURCE_HSI</code></li> <li>- <code>LL_RCC_I2C1_CLKSOURCE_SYSCLK</code></li> <li>- <code>LL_RCC_I2C2_CLKSOURCE_HSI</code> (*)</li> <li>- <code>LL_RCC_I2C2_CLKSOURCE_SYSCLK</code> (*)</li> <li>- <code>LL_RCC_I2C3_CLKSOURCE_HSI</code> (*)</li> <li>- <code>LL_RCC_I2C3_CLKSOURCE_SYSCLK</code> (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR3 I2C1SW <code>LL_RCC_SetI2CClockSource</code></li> <li>• CFGR3 I2C2SW <code>LL_RCC_SetI2CClockSource</code></li> <li>• CFGR3 I2C3SW <code>LL_RCC_SetI2CClockSource</code></li> </ul>                                                                                                                                                                                                                                                                                                             |

**LL\_RCC\_SetI2SClockSource**

|                                                   |                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_SetI2SClockSource (uint32_t I2SxSource)</code>                                                                                                                                                                                         |
| Function description                              | Configure I2Sx clock source.                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2SxSource:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_RCC_I2S_CLKSOURCE_SYSCLK</code></li> <li>- <code>LL_RCC_I2S_CLKSOURCE_PIN</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR I2SSRC <code>LL_RCC_SetI2SClockSource</code></li> </ul>                                                                                                                                                                    |

**LL\_RCC\_SetTIMClockSource**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RCC_SetTIMClockSource (uint32_t TIMxSource)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Configure TIMx clock source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMxSource:</b> This parameter can be one of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- <code>LL_RCC_TIM1_CLKSOURCE_PCLK2</code></li> <li>- <code>LL_RCC_TIM1_CLKSOURCE_PLL</code></li> <li>- <code>LL_RCC_TIM8_CLKSOURCE_PCLK2</code> (*)</li> <li>- <code>LL_RCC_TIM8_CLKSOURCE_PLL</code> (*)</li> <li>- <code>LL_RCC_TIM15_CLKSOURCE_PCLK2</code> (*)</li> <li>- <code>LL_RCC_TIM15_CLKSOURCE_PLL</code> (*)</li> <li>- <code>LL_RCC_TIM16_CLKSOURCE_PCLK2</code> (*)</li> <li>- <code>LL_RCC_TIM16_CLKSOURCE_PLL</code> (*)</li> <li>- <code>LL_RCC_TIM17_CLKSOURCE_PCLK2</code> (*)</li> <li>- <code>LL_RCC_TIM17_CLKSOURCE_PLL</code> (*)</li> </ul> </li> </ul> |

- LL\_RCC\_TIM20\_CLKSOURCE\_PCLK2 (\*)
- LL\_RCC\_TIM20\_CLKSOURCE\_PLL (\*)
- LL\_RCC\_TIM2\_CLKSOURCE\_PCLK1 (\*)
- LL\_RCC\_TIM2\_CLKSOURCE\_PLL (\*)
- LL\_RCC\_TIM34\_CLKSOURCE\_PCLK1 (\*)
- LL\_RCC\_TIM34\_CLKSOURCE\_PLL (\*)

**Return values**

- **None**

**Reference Manual to  
LL API cross  
reference:**

- CFGR3 TIM1SW LL\_RCC\_SetTIMClockSource
- CFGR3 TIM8SW LL\_RCC\_SetTIMClockSource
- CFGR3 TIM15SW LL\_RCC\_SetTIMClockSource
- CFGR3 TIM16SW LL\_RCC\_SetTIMClockSource
- CFGR3 TIM17SW LL\_RCC\_SetTIMClockSource
- CFGR3 TIM20SW LL\_RCC\_SetTIMClockSource
- CFGR3 TIM2SW LL\_RCC\_SetTIMClockSource
- CFGR3 TIM34SW LL\_RCC\_SetTIMClockSource

**LL\_RCC\_SetUSBClockSource**

**Function name** `__STATIC_INLINE void LL_RCC_SetUSBClockSource(uint32_t USBxSource)`

**Function description** Configure USB clock source.

**Parameters**

- **USBxSource:** This parameter can be one of the following values:
  - LL\_RCC\_USB\_CLKSOURCE\_PLL
  - LL\_RCC\_USB\_CLKSOURCE\_PLL\_DIV\_1\_5

**Return values**

- **None**

**Reference Manual to  
LL API cross  
reference:**

- CFGR USBPRE LL\_RCC\_SetUSBClockSource

**LL\_RCC\_SetADCClockSource**

**Function name** `__STATIC_INLINE void LL_RCC_SetADCClockSource(uint32_t ADCxSource)`

**Function description** Configure ADC clock source.

**Parameters**

- **ADCxSource:** This parameter can be one of the following values: (\*) value not defined in all devices.
  - LL\_RCC\_ADC12\_CLKSRC\_HCLK
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_1
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_2
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_4
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_6
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_8
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_10
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_12
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_16
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_32
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_64
  - LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_128

- LL\_RCC\_ADC12\_CLKSRC\_PLL\_DIV\_256
- LL\_RCC\_ADC34\_CLKSRC\_HCLK (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_1 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_2 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_4 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_6 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_8 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_10 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_12 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_16 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_32 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_64 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_128 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_256 (\*)

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CFGR2 ADCPRE12 LL\_RCC\_SetADCClockSource
- CFGR2 ADCPRE34 LL\_RCC\_SetADCClockSource

### **LL\_RCC\_GetUSARTClockSource**

Function name      **`_STATIC_INLINE uint32_t LL_RCC_GetUSARTClockSource  
(uint32_t USARTx)`**

Function description

Get USARTx clock source.

Parameters

- **USARTx:** This parameter can be one of the following values:  
(\*) value not defined in all devices.
  - LL\_RCC\_USART1\_CLKSOURCE
  - LL\_RCC\_USART2\_CLKSOURCE (\*)
  - LL\_RCC\_USART3\_CLKSOURCE (\*)

Return values

- **Returned:** value can be one of the following values: (\*)  
value not defined in all devices.
  - LL\_RCC\_USART1\_CLKSOURCE\_PCLK1 (\*)
  - LL\_RCC\_USART1\_CLKSOURCE\_PCLK2 (\*)
  - LL\_RCC\_USART1\_CLKSOURCE\_SYSCLK
  - LL\_RCC\_USART1\_CLKSOURCE\_LSE
  - LL\_RCC\_USART1\_CLKSOURCE\_HSI
  - LL\_RCC\_USART2\_CLKSOURCE\_PCLK1 (\*)
  - LL\_RCC\_USART2\_CLKSOURCE\_SYSCLK (\*)
  - LL\_RCC\_USART2\_CLKSOURCE\_LSE (\*)
  - LL\_RCC\_USART2\_CLKSOURCE\_HSI (\*)
  - LL\_RCC\_USART3\_CLKSOURCE\_PCLK1 (\*)
  - LL\_RCC\_USART3\_CLKSOURCE\_SYSCLK (\*)
  - LL\_RCC\_USART3\_CLKSOURCE\_LSE (\*)
  - LL\_RCC\_USART3\_CLKSOURCE\_HSI (\*)

Reference Manual to  
LL API cross  
reference:

- CFGR3 USART1SW LL\_RCC\_GetUSARTClockSource
- CFGR3 USART2SW LL\_RCC\_GetUSARTClockSource
- CFGR3 USART3SW LL\_RCC\_GetUSARTClockSource

**LL\_RCC\_GetUARTClockSource**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_GetUARTClockSource<br/>(uint32_t UARTx)</code>                                                                                                                                                                                                                                                                                                                                                                                                         |
| Function description                              | Get UARTx clock source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>UARTx:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RCC_UART4_CLKSOURCE</li> <li>- LL_RCC_UART5_CLKSOURCE</li> </ul> </li> </ul>                                                                                                                                                                                                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RCC_UART4_CLKSOURCE_PCLK1</li> <li>- LL_RCC_UART4_CLKSOURCE_SYSCLK</li> <li>- LL_RCC_UART4_CLKSOURCE_LSE</li> <li>- LL_RCC_UART4_CLKSOURCE_HSI</li> <li>- LL_RCC_UART5_CLKSOURCE_PCLK1</li> <li>- LL_RCC_UART5_CLKSOURCE_SYSCLK</li> <li>- LL_RCC_UART5_CLKSOURCE_LSE</li> <li>- LL_RCC_UART5_CLKSOURCE_HSI</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR3 UART4SW LL_RCC_GetUARTClockSource</li> <li>• CFGR3 UART5SW LL_RCC_GetUARTClockSource</li> </ul>                                                                                                                                                                                                                                                                                                                                               |

**LL\_RCC\_GetI2CClockSource**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_GetI2CClockSource<br/>(uint32_t I2Cx)</code>                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description                              | Get I2Cx clock source.                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>I2Cx:</b> This parameter can be one of the following values: (*)<br/>value not defined in all devices.           <ul style="list-style-type: none"> <li>- LL_RCC_I2C1_CLKSOURCE</li> <li>- LL_RCC_I2C2_CLKSOURCE (*)</li> <li>- LL_RCC_I2C3_CLKSOURCE (*)</li> </ul> </li> </ul>                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (*)<br/>value not defined in all devices.           <ul style="list-style-type: none"> <li>- LL_RCC_I2C1_CLKSOURCE_HSI</li> <li>- LL_RCC_I2C1_CLKSOURCE_SYSCLK</li> <li>- LL_RCC_I2C2_CLKSOURCE_HSI (*)</li> <li>- LL_RCC_I2C2_CLKSOURCE_SYSCLK (*)</li> <li>- LL_RCC_I2C3_CLKSOURCE_HSI (*)</li> <li>- LL_RCC_I2C3_CLKSOURCE_SYSCLK (*)</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR3 I2C1SW LL_RCC_GetI2CClockSource</li> <li>• CFGR3 I2C2SW LL_RCC_GetI2CClockSource</li> <li>• CFGR3 I2C3SW LL_RCC_GetI2CClockSource</li> </ul>                                                                                                                                                                                                                                                                         |

**LL\_RCC\_GetI2SClockSource**

|                      |                                                                                    |
|----------------------|------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_RCC_GetI2SClockSource<br/>(uint32_t I2Sx)</code> |
| Function description | Get I2Sx clock source.                                                             |

|                                                   |                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>I2Sx:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RCC_I2S_CLKSOURCE</li> </ul> </li> </ul>                                       |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RCC_I2S_CLKSOURCE_SYSCLK</li> <li>- LL_RCC_I2S_CLKSOURCE_PIN</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CFGR I2SSRC LL_RCC_GetI2SClockSource</li> </ul>                                                                                                                                                      |

### LL\_RCC\_GetTIMClockSource

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_GetTIMClockSource(<br/>          uint32_t TIMx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description                              | Get TIMx clock source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>TIMx:</b> This parameter can be one of the following values: (*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- LL_RCC_TIM1_CLKSOURCE</li> <li>- LL_RCC_TIM2_CLKSOURCE (*)</li> <li>- LL_RCC_TIM8_CLKSOURCE (*)</li> <li>- LL_RCC_TIM15_CLKSOURCE (*)</li> <li>- LL_RCC_TIM16_CLKSOURCE (*)</li> <li>- LL_RCC_TIM17_CLKSOURCE (*)</li> <li>- LL_RCC_TIM20_CLKSOURCE (*)</li> <li>- LL_RCC_TIM34_CLKSOURCE (*)</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values: (*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- LL_RCC_TIM1_CLKSOURCE_PCLK2</li> <li>- LL_RCC_TIM1_CLKSOURCE_PLL</li> <li>- LL_RCC_TIM8_CLKSOURCE_PCLK2 (*)</li> <li>- LL_RCC_TIM8_CLKSOURCE_PLL (*)</li> <li>- LL_RCC_TIM15_CLKSOURCE_PCLK2 (*)</li> <li>- LL_RCC_TIM15_CLKSOURCE_PLL (*)</li> <li>- LL_RCC_TIM16_CLKSOURCE_PCLK2 (*)</li> <li>- LL_RCC_TIM16_CLKSOURCE_PLL (*)</li> <li>- LL_RCC_TIM17_CLKSOURCE_PCLK2 (*)</li> <li>- LL_RCC_TIM17_CLKSOURCE_PLL (*)</li> <li>- LL_RCC_TIM20_CLKSOURCE_PCLK2 (*)</li> <li>- LL_RCC_TIM20_CLKSOURCE_PLL (*)</li> <li>- LL_RCC_TIM2_CLKSOURCE_PCLK1 (*)</li> <li>- LL_RCC_TIM2_CLKSOURCE_PLL (*)</li> <li>- LL_RCC_TIM34_CLKSOURCE_PCLK1 (*)</li> <li>- LL_RCC_TIM34_CLKSOURCE_PLL (*)</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CFGR3 TIM1SW LL_RCC_GetTIMClockSource</li> <li>CFGR3 TIM8SW LL_RCC_GetTIMClockSource</li> <li>CFGR3 TIM15SW LL_RCC_GetTIMClockSource</li> <li>CFGR3 TIM16SW LL_RCC_GetTIMClockSource</li> <li>CFGR3 TIM17SW LL_RCC_GetTIMClockSource</li> <li>CFGR3 TIM20SW LL_RCC_GetTIMClockSource</li> <li>CFGR3 TIM2SW LL_RCC_GetTIMClockSource</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

- CFGR3 TIM34SW LL\_RCC\_GetTIMClockSource

### **LL\_RCC\_GetUSBClockSource**

|                                                   |                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_RCC_GetUSBClockSource(uint32_t USBx)</code></b>                                                                                                                                                                |
| Function description                              | Get USBx clock source.                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USBx:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RCC_USB_CLKSOURCE</li> </ul> </li> </ul>                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RCC_USB_CLKSOURCE_PLL</li> <li>- LL_RCC_USB_CLKSOURCE_PLL_DIV_1_5</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFGR USBPRE LL_RCC_GetUSBClockSource</li> </ul>                                                                                                                                                           |

### **LL\_RCC\_GetADCClockSource**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_RCC_GetADCClockSource(uint32_t ADCx)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description | Get ADCx clock source.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>ADCx:</b> This parameter can be one of the following values:<br/>(*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- LL_RCC_ADC12_CLKSOURCE</li> <li>- LL_RCC_ADC34_CLKSOURCE (*)</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (*)<br/>value not defined in all devices.           <ul style="list-style-type: none"> <li>- LL_RCC_ADC12_CLKSRC_HCLK</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_1</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_2</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_4</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_6</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_8</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_10</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_12</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_16</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_32</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_64</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_128</li> <li>- LL_RCC_ADC12_CLKSRC_PLL_DIV_256</li> <li>- LL_RCC_ADC34_CLKSRC_HCLK (*)</li> <li>- LL_RCC_ADC34_CLKSRC_PLL_DIV_1 (*)</li> <li>- LL_RCC_ADC34_CLKSRC_PLL_DIV_2 (*)</li> <li>- LL_RCC_ADC34_CLKSRC_PLL_DIV_4 (*)</li> <li>- LL_RCC_ADC34_CLKSRC_PLL_DIV_6 (*)</li> <li>- LL_RCC_ADC34_CLKSRC_PLL_DIV_8 (*)</li> <li>- LL_RCC_ADC34_CLKSRC_PLL_DIV_10 (*)</li> <li>- LL_RCC_ADC34_CLKSRC_PLL_DIV_12 (*)</li> <li>- LL_RCC_ADC34_CLKSRC_PLL_DIV_16 (*)</li> </ul> </li> </ul> |

- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_32 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_64 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_128 (\*)
- LL\_RCC\_ADC34\_CLKSRC\_PLL\_DIV\_256 (\*)

Reference Manual to  
LL API cross  
reference:

- CFGR2 ADCPRE12 LL\_RCC\_GetADCClockSource
- CFGR2 ADCPRE34 LL\_RCC\_GetADCClockSource

### **LL\_RCC\_SetRTCClockSource**

|                                                   |                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_SetRTCClockSource(<br/>  uint32_t Source)</code>                                                                                                                                                                                                                                               |
| Function description                              | Set RTC Clock Source.                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Source:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RCC_RTC_CLKSOURCE_NONE</li> <li>- LL_RCC_RTC_CLKSOURCE_LSE</li> <li>- LL_RCC_RTC_CLKSOURCE_LSI</li> <li>- LL_RCC_RTC_CLKSOURCE_HSE_DIV32</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• Once the RTC clock source has been selected, it cannot be changed any more unless the Backup domain is reset. The BDRST bit can be used to reset them.</li> </ul>                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDCR RTCSEL LL_RCC_SetRTCClockSource</li> </ul>                                                                                                                                                                                                                                         |

### **LL\_RCC\_GetRTCClockSource**

|                                                   |                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_GetRTCClockSource(<br/>  void )</code>                                                                                                                                                                                                                                              |
| Function description                              | Get RTC Clock Source.                                                                                                                                                                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RCC_RTC_CLKSOURCE_NONE</li> <li>- LL_RCC_RTC_CLKSOURCE_LSE</li> <li>- LL_RCC_RTC_CLKSOURCE_LSI</li> <li>- LL_RCC_RTC_CLKSOURCE_HSE_DIV32</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDCR RTCSEL LL_RCC_GetRTCClockSource</li> </ul>                                                                                                                                                                                                                                  |

**LL\_RCC\_EnableRTC**

|                                                   |                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_EnableRTC (void )</code></b>             |
| Function description                              | Enable RTC.                                                                   |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• BDCR RTCEN LL_RCC_EnableRTC</li></ul> |

**LL\_RCC\_DisableRTC**

|                                                   |                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_DisableRTC (void )</code></b>             |
| Function description                              | Disable RTC.                                                                   |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• BDCR RTCEN LL_RCC_DisableRTC</li></ul> |

**LL\_RCC\_IsEnabledRTC**

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RCC_IsEnabledRTC (void )</code></b>         |
| Function description                              | Check if RTC has been enabled or not.                                            |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• BDCR RTCEN LL_RCC_IsEnabledRTC</li></ul> |

**LL\_RCC\_ForceBackupDomainReset**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_ForceBackupDomainReset (void )</code></b>             |
| Function description                              | Force the Backup domain reset.                                                             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• BDCR BDRST LL_RCC_ForceBackupDomainReset</li></ul> |

**LL\_RCC\_ReleaseBackupDomainReset**

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_ReleaseBackupDomainReset (void )</code></b>             |
| Function description                              | Release the Backup domain reset.                                                             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• BDCR BDRST LL_RCC_ReleaseBackupDomainReset</li></ul> |

**LL\_RCC\_PLL\_Enable**

Function name **`__STATIC_INLINE void LL_RCC_PLL_Enable (void )`**

Function description Enable PLL.

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
CR PLLON LL\_RCC\_PLL\_Enable

**LL\_RCC\_PLL\_Disable**

Function name **`__STATIC_INLINE void LL_RCC_PLL_Disable (void )`**

Function description Disable PLL.

Return values • **None**

Notes • Cannot be disabled if the PLL clock is used as the system clock

Reference Manual to  
LL API cross  
reference:  
CR PLLON LL\_RCC\_PLL\_Disable

**LL\_RCC\_PLL\_IsReady**

Function name **`__STATIC_INLINE uint32_t LL_RCC_PLL_IsReady (void )`**

Function description Check if PLL Ready.

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
CR PLLRDY LL\_RCC\_PLL\_IsReady

**LL\_RCC\_PLL\_ConfigDomain\_SYS**

Function name **`__STATIC_INLINE void LL_RCC_PLL_ConfigDomain_SYS  
(uint32_t Source, uint32_t PLLMul)`**

Function description Configure PLL used for SYSCLK Domain.

Parameters • **Source:** This parameter can be one of the following values:

- `LL_RCC_PLLSOURCE_HSI_DIV_2`
- `LL_RCC_PLLSOURCE_HSE_DIV_1`
- `LL_RCC_PLLSOURCE_HSE_DIV_2`
- `LL_RCC_PLLSOURCE_HSE_DIV_3`
- `LL_RCC_PLLSOURCE_HSE_DIV_4`
- `LL_RCC_PLLSOURCE_HSE_DIV_5`
- `LL_RCC_PLLSOURCE_HSE_DIV_6`
- `LL_RCC_PLLSOURCE_HSE_DIV_7`
- `LL_RCC_PLLSOURCE_HSE_DIV_8`
- `LL_RCC_PLLSOURCE_HSE_DIV_9`
- `LL_RCC_PLLSOURCE_HSE_DIV_10`
- `LL_RCC_PLLSOURCE_HSE_DIV_11`
- `LL_RCC_PLLSOURCE_HSE_DIV_12`

- LL\_RCC\_PLLSOURCE\_HSE\_DIV\_13
- LL\_RCC\_PLLSOURCE\_HSE\_DIV\_14
- LL\_RCC\_PLLSOURCE\_HSE\_DIV\_15
- LL\_RCC\_PLLSOURCE\_HSE\_DIV\_16
- **PLLMul:** This parameter can be one of the following values:
  - LL\_RCC\_PLL\_MUL\_2
  - LL\_RCC\_PLL\_MUL\_3
  - LL\_RCC\_PLL\_MUL\_4
  - LL\_RCC\_PLL\_MUL\_5
  - LL\_RCC\_PLL\_MUL\_6
  - LL\_RCC\_PLL\_MUL\_7
  - LL\_RCC\_PLL\_MUL\_8
  - LL\_RCC\_PLL\_MUL\_9
  - LL\_RCC\_PLL\_MUL\_10
  - LL\_RCC\_PLL\_MUL\_11
  - LL\_RCC\_PLL\_MUL\_12
  - LL\_RCC\_PLL\_MUL\_13
  - LL\_RCC\_PLL\_MUL\_14
  - LL\_RCC\_PLL\_MUL\_15
  - LL\_RCC\_PLL\_MUL\_16

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CFGR PLLSRC LL\_RCC\_PLL\_ConfigDomain\_SYS
- CFGR PLLMUL LL\_RCC\_PLL\_ConfigDomain\_SYS
- CFGR2 PREDIV LL\_RCC\_PLL\_ConfigDomain\_SYS

### **LL\_RCC\_PLL\_GetMainSource**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_RCC\_PLL\_GetMainSource(void )**

Function description

Get the oscillator used as PLL clock source.

Return values

- **Returned:** value can be one of the following values: (\*)  
value not defined in all devices
  - LL\_RCC\_PLLSOURCE\_HSI (\*)
  - LL\_RCC\_PLLSOURCE\_HSI\_DIV\_2 (\*)
  - LL\_RCC\_PLLSOURCE\_HSE

Reference Manual to  
LL API cross  
reference:

- CFGR PLLSRC LL\_RCC\_PLL\_GetMainSource

### **LL\_RCC\_PLL\_GetMultiplicator**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_RCC\_PLL\_GetMultiplicator(void )**

Function description

Get PLL multiplication Factor.

Return values

- **Returned:** value can be one of the following values:
  - LL\_RCC\_PLL\_MUL\_2
  - LL\_RCC\_PLL\_MUL\_3
  - LL\_RCC\_PLL\_MUL\_4
  - LL\_RCC\_PLL\_MUL\_5

- LL\_RCC\_PLL\_MUL\_6
- LL\_RCC\_PLL\_MUL\_7
- LL\_RCC\_PLL\_MUL\_8
- LL\_RCC\_PLL\_MUL\_9
- LL\_RCC\_PLL\_MUL\_10
- LL\_RCC\_PLL\_MUL\_11
- LL\_RCC\_PLL\_MUL\_12
- LL\_RCC\_PLL\_MUL\_13
- LL\_RCC\_PLL\_MUL\_14
- LL\_RCC\_PLL\_MUL\_15
- LL\_RCC\_PLL\_MUL\_16

Reference Manual to  
LL API cross  
reference:

- CFGR PLLMUL LL\_RCC\_PLL\_GetMultiplicator

### **LL\_RCC\_PLL\_GetPrediv**

Function name **`_STATIC_INLINE uint32_t LL_RCC_PLL_GetPrediv (void )`**

Function description Get PREDIV division factor for the main PLL.

Return values

- **Returned:** value can be one of the following values:
  - LL\_RCC\_PREDIV\_DIV\_1
  - LL\_RCC\_PREDIV\_DIV\_2
  - LL\_RCC\_PREDIV\_DIV\_3
  - LL\_RCC\_PREDIV\_DIV\_4
  - LL\_RCC\_PREDIV\_DIV\_5
  - LL\_RCC\_PREDIV\_DIV\_6
  - LL\_RCC\_PREDIV\_DIV\_7
  - LL\_RCC\_PREDIV\_DIV\_8
  - LL\_RCC\_PREDIV\_DIV\_9
  - LL\_RCC\_PREDIV\_DIV\_10
  - LL\_RCC\_PREDIV\_DIV\_11
  - LL\_RCC\_PREDIV\_DIV\_12
  - LL\_RCC\_PREDIV\_DIV\_13
  - LL\_RCC\_PREDIV\_DIV\_14
  - LL\_RCC\_PREDIV\_DIV\_15
  - LL\_RCC\_PREDIV\_DIV\_16

Notes

- They can be written only when the PLL is disabled

Reference Manual to  
LL API cross  
reference:

- CFGR2 PREDIV LL\_RCC\_PLL\_GetPrediv

### **LL\_RCC\_ClearFlag\_LSIRDY**

Function name **`_STATIC_INLINE void LL_RCC_ClearFlag_LSIRDY (void )`**

Function description Clear LSI ready interrupt flag.

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CIR LSIRDYC LL\_RCC\_ClearFlag\_LSIRDY

**LL\_RCC\_ClearFlag\_LSERDY**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_ClearFlag_LSERDY (void )</code></b>              |
| Function description                              | Clear LSE ready interrupt flag.                                                       |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CIR LSERDYC LL_RCC_ClearFlag_LSERDY</li></ul> |

**LL\_RCC\_ClearFlag\_HSIRDY**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_ClearFlag_HSIRDY (void )</code></b>              |
| Function description                              | Clear HSI ready interrupt flag.                                                       |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CIR HSIRDYC LL_RCC_ClearFlag_HSIRDY</li></ul> |

**LL\_RCC\_ClearFlag\_HSERDY**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_ClearFlag_HSERDY (void )</code></b>              |
| Function description                              | Clear HSE ready interrupt flag.                                                       |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CIR HSERDYC LL_RCC_ClearFlag_HSERDY</li></ul> |

**LL\_RCC\_ClearFlag\_PLLRDY**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_ClearFlag_PLLRDY (void )</code></b>              |
| Function description                              | Clear PLL ready interrupt flag.                                                       |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CIR PLLRDYC LL_RCC_ClearFlag_PLLRDY</li></ul> |

**LL\_RCC\_ClearFlag\_HSECSS**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_ClearFlag_HSECSS (void )</code></b>           |
| Function description                              | Clear Clock security system interrupt flag.                                        |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CIR CSSC LL_RCC_ClearFlag_HSECSS</li></ul> |

**LL\_RCC\_IsActiveFlag\_LSIRDY**

Function name      `__STATIC_INLINE uint32_t LL_RCC_IsActiveFlag_LSIRDY(void)`

Function description      Check if LSI ready interrupt occurred or not.

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
CIR LSIRDYF LL\_RCC\_IsActiveFlag\_LSIRDY

**LL\_RCC\_IsActiveFlag\_LSERDY**

Function name      `__STATIC_INLINE uint32_t LL_RCC_IsActiveFlag_LSERDY(void)`

Function description      Check if LSE ready interrupt occurred or not.

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
CIR LSERDYF LL\_RCC\_IsActiveFlag\_LSERDY

**LL\_RCC\_IsActiveFlag\_HSIRDY**

Function name      `__STATIC_INLINE uint32_t LL_RCC_IsActiveFlag_HSIRDY(void)`

Function description      Check if HSI ready interrupt occurred or not.

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
CIR HSIRDYF LL\_RCC\_IsActiveFlag\_HSIRDY

**LL\_RCC\_IsActiveFlag\_HSERDY**

Function name      `__STATIC_INLINE uint32_t LL_RCC_IsActiveFlag_HSERDY(void)`

Function description      Check if HSE ready interrupt occurred or not.

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
CIR HSERDYF LL\_RCC\_IsActiveFlag\_HSERDY

**LL\_RCC\_IsActiveFlag\_MCO1**

Function name      `__STATIC_INLINE uint32_t LL_RCC_IsActiveFlag_MCO1(void)`

Function description      Check if switch to new MCO source is effective or not.

Return values      • **State:** of bit (1 or 0).

- Reference Manual to  
LL API cross  
reference:
- CFGR MCOF LL\_RCC\_IsActiveFlag\_MCO1

### **LL\_RCC\_IsActiveFlag\_PLLRDY**

- Function name            **\_STATIC\_INLINE uint32\_t LL\_RCC\_IsActiveFlag\_PLLRDY(void)**
- Function description    Check if PLL ready interrupt occurred or not.
- Return values            • **State:** of bit (1 or 0).
- Reference Manual to  
LL API cross  
reference:
- CIR PLLRDYF LL\_RCC\_IsActiveFlag\_PLLRDY

### **LL\_RCC\_IsActiveFlag\_HSECSS**

- Function name            **\_STATIC\_INLINE uint32\_t LL\_RCC\_IsActiveFlag\_HSECSS(void)**
- Function description    Check if Clock security system interrupt occurred or not.
- Return values            • **State:** of bit (1 or 0).
- Reference Manual to  
LL API cross  
reference:
- CIR CSSF LL\_RCC\_IsActiveFlag\_HSECSS

### **LL\_RCC\_IsActiveFlag\_IWDGRST**

- Function name            **\_STATIC\_INLINE uint32\_t LL\_RCC\_IsActiveFlag\_IWDGRST(void)**
- Function description    Check if RCC flag Independent Watchdog reset is set or not.
- Return values            • **State:** of bit (1 or 0).
- Reference Manual to  
LL API cross  
reference:
- CSR IWDGRSTF LL\_RCC\_IsActiveFlag\_IWDGRST

### **LL\_RCC\_IsActiveFlag\_LPWRRST**

- Function name            **\_STATIC\_INLINE uint32\_t LL\_RCC\_IsActiveFlag\_LPWRRST(void)**
- Function description    Check if RCC flag Low Power reset is set or not.
- Return values            • **State:** of bit (1 or 0).
- Reference Manual to  
LL API cross  
reference:
- CSR LPWRRSTF LL\_RCC\_IsActiveFlag\_LPWRRST

### **LL\_RCC\_IsActiveFlag\_OBLRST**

- Function name            **\_STATIC\_INLINE uint32\_t LL\_RCC\_IsActiveFlag\_OBLRST(void)**

---

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function description                              | Check if RCC flag is set or not.                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | CSR OBLRSTF LL_RCC_IsActiveFlag_OBLRST                                             |

### LL\_RCC\_IsActiveFlag\_PINRST

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_IsActiveFlag_PINRST(<br/>void )</code>       |
| Function description                              | Check if RCC flag Pin reset is set or not.                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | CSR PINRSTF LL_RCC_IsActiveFlag_PINRST                                             |

### LL\_RCC\_IsActiveFlag\_PORRST

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_IsActiveFlag_PORRST(<br/>void )</code>       |
| Function description                              | Check if RCC flag POR/PDR reset is set or not.                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | CSR PORRSTF LL_RCC_IsActiveFlag_PORRST                                             |

### LL\_RCC\_IsActiveFlag\_SFTRST

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_IsActiveFlag_SFTRST(<br/>void )</code>       |
| Function description                              | Check if RCC flag Software reset is set or not.                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | CSR SFTRSTF LL_RCC_IsActiveFlag_SFTRST                                             |

### LL\_RCC\_IsActiveFlag\_WWDGRST

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_IsActiveFlag_WWDGRST(<br/>void )</code>      |
| Function description                              | Check if RCC flag Window Watchdog reset is set or not.                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | CSR WWDGRSTF LL_RCC_IsActiveFlag_WWDGRST                                           |

**LL\_RCC\_IsActiveFlag\_V18PWRRST**

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_RCC_IsActiveFlag_V18PWRRST (void )</code>  |
| Function description                              | Check if RCC Reset flag of the 1.8 V domain is set or not.                       |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | • CSR V18PWRRSTF LL_RCC_IsActiveFlag_V18PWRRST                                   |

**LL\_RCC\_ClearResetFlags**

|                                                   |                                                                  |
|---------------------------------------------------|------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_ClearResetFlags (void )</code> |
| Function description                              | Set RMVF bit to clear the reset flags.                           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>    |
| Reference Manual to<br>LL API cross<br>reference: | • CSR RMVF LL_RCC_ClearResetFlags                                |

**LL\_RCC\_EnableIT\_LSIRDY**

|                                                   |                                                                  |
|---------------------------------------------------|------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_EnableIT_LSIRDY (void )</code> |
| Function description                              | Enable LSI ready interrupt.                                      |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>    |
| Reference Manual to<br>LL API cross<br>reference: | • CIR LSIRDYIE LL_RCC_EnableIT_LSIRDY                            |

**LL\_RCC\_EnableIT\_LSERDY**

|                                                   |                                                                  |
|---------------------------------------------------|------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_EnableIT_LSERDY (void )</code> |
| Function description                              | Enable LSE ready interrupt.                                      |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>    |
| Reference Manual to<br>LL API cross<br>reference: | • CIR LSERDYIE LL_RCC_EnableIT_LSERDY                            |

**LL\_RCC\_EnableIT\_HSIRDY**

|                                                   |                                                                  |
|---------------------------------------------------|------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RCC_EnableIT_HSIRDY (void )</code> |
| Function description                              | Enable HSI ready interrupt.                                      |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>    |
| Reference Manual to<br>LL API cross<br>reference: | • CIR HSIRDYIE LL_RCC_EnableIT_HSIRDY                            |

**LL\_RCC\_EnableIT\_HSERDY**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_EnableIT_HSERDY (void )</code></b>               |
| Function description                              | Enable HSE ready interrupt.                                                           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CIR HSERDYIE LL_RCC_EnableIT_HSERDY</li></ul> |

**LL\_RCC\_EnableIT\_PLLRDY**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_EnableIT_PLLRDY (void )</code></b>               |
| Function description                              | Enable PLL ready interrupt.                                                           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CIR PLLRDYIE LL_RCC_EnableIT_PLLRDY</li></ul> |

**LL\_RCC\_DisableIT\_LSIRDY**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_DisableIT_LSIRDY (void )</code></b>               |
| Function description                              | Disable LSI ready interrupt.                                                           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CIR LSIRDYIE LL_RCC_DisableIT_LSIRDY</li></ul> |

**LL\_RCC\_DisableIT\_LSERDY**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_DisableIT_LSERDY (void )</code></b>               |
| Function description                              | Disable LSE ready interrupt.                                                           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CIR LSERDYIE LL_RCC_DisableIT_LSERDY</li></ul> |

**LL\_RCC\_DisableIT\_HSIRDY**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_DisableIT_HSIRDY (void )</code></b>               |
| Function description                              | Disable HSI ready interrupt.                                                           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CIR HSIRDYIE LL_RCC_DisableIT_HSIRDY</li></ul> |

**LL\_RCC\_DisableIT\_HSERDY**

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_DisableIT_HSERDY (void )</code></b>                 |
| Function description                              | Disable HSE ready interrupt.                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CIR HSERDYIE LL_RCC_DisableIT_HSERDY</li> </ul> |

**LL\_RCC\_DisableIT\_PLLRDY**

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RCC_DisableIT_PLLRDY (void )</code></b>                 |
| Function description                              | Disable PLL ready interrupt.                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CIR PLLRDYIE LL_RCC_DisableIT_PLLRDY</li> </ul> |

**LL\_RCC\_IsEnabledIT\_LSIRDY**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RCC_IsEnabledIT_LSIRDY (void )</code></b>             |
| Function description                              | Checks if LSI ready interrupt source is enabled or disabled.                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CIR LSIRDYIE LL_RCC_IsEnabledIT_LSIRDY</li> </ul> |

**LL\_RCC\_IsEnabledIT\_LSERDY**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RCC_IsEnabledIT_LSERDY (void )</code></b>             |
| Function description                              | Checks if LSE ready interrupt source is enabled or disabled.                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CIR LSERDYIE LL_RCC_IsEnabledIT_LSERDY</li> </ul> |

**LL\_RCC\_IsEnabledIT\_HSIRDY**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RCC_IsEnabledIT_HSIRDY (void )</code></b>             |
| Function description                              | Checks if HSI ready interrupt source is enabled or disabled.                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CIR HSIRDYIE LL_RCC_IsEnabledIT_HSIRDY</li> </ul> |

**LL\_RCC\_IsEnabledIT\_HSERDY**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_IsEnabledIT_HSERDY(void)</code>              |
| Function description                              | Checks if HSE ready interrupt source is enabled or disabled.                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | • CIR HSERDYIE LL_RCC_IsEnabledIT_HSERDY                                           |

**LL\_RCC\_IsEnabledIT\_PLLRDY**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RCC_IsEnabledIT_PLLRDY(void)</code>              |
| Function description                              | Checks if PLL ready interrupt source is enabled or disabled.                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | • CIR PLLRDYIE LL_RCC_IsEnabledIT_PLLRDY                                           |

**LL\_RCC\_DelInit**

|                      |                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_RCC_DelInit(void)</code>                                                                                                                                                                                                                                                                                                                   |
| Function description | Reset the RCC clock configuration to the default reset state.                                                                                                                                                                                                                                                                                                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: RCC registers are de-initialized</li> <li>– ERROR: not applicable</li> </ul> </li> </ul>                                                                                                                         |
| Notes                | <ul style="list-style-type: none"> <li>• The default reset state of the clock configuration is given below: HSI ON and used as system clock sourceHSE and PLL OFFAHB, APB1 and APB2 prescaler set to 1.CSS, MCO OFFAll interrupts disabled</li> <li>• This function doesn't modify the configuration of the Peripheral clocksLSI, LSE and RTC clocks</li> </ul> |

**LL\_RCC\_GetSystemClocksFreq**

|                      |                                                                                                                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void LL_RCC_GetSystemClocksFreq(LL_RCC_ClocksTypeDef * RCC_Clocks)</code>                                                                                                                                                               |
| Function description | Return the frequencies of different on chip clocks; System, AHB, APB1 and APB2 buses clocks.                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RCC_Clocks:</b> pointer to a LL_RCC_ClocksTypeDef structure which will hold the clocks frequencies</li> </ul>                                                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>• Each time SYSCLK, HCLK, PCLK1 and/or PCLK2 clock changes, this function must be called to update structure fields. Otherwise, any configuration based on this function will be incorrect.</li> </ul> |

**LL\_RCC\_GetUSARTClockFreq**

|                      |                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t LL_RCC_GetUSARTClockFreq (uint32_t USARTxSource)</code>                                                                                                                                                                                                                                                                                                        |
| Function description | Return USARTx clock frequency.                                                                                                                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTxSource:</b> This parameter can be one of the following values: (*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- <code>LL_RCC_USART1_CLKSOURCE</code></li> <li>- <code>LL_RCC_USART2_CLKSOURCE</code> (*)</li> <li>- <code>LL_RCC_USART3_CLKSOURCE</code> (*)</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>USART:</b> clock frequency (in Hz)           <ul style="list-style-type: none"> <li>- <code>LL_RCC_PERIPH_FREQUENCY_NO</code> indicates that oscillator (HSI or LSE) is not ready</li> </ul> </li> </ul>                                                                                                                          |

**LL\_RCC\_GetUARTClockFreq**

|                      |                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t LL_RCC_GetUARTClockFreq (uint32_t UARTxSource)</code>                                                                                                                                                                                                         |
| Function description | Return UARTx clock frequency.                                                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>UARTxSource:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- <code>LL_RCC_UART4_CLKSOURCE</code></li> <li>- <code>LL_RCC_UART5_CLKSOURCE</code></li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>UART:</b> clock frequency (in Hz)           <ul style="list-style-type: none"> <li>- <code>LL_RCC_PERIPH_FREQUENCY_NO</code> indicates that oscillator (HSI or LSE) is not ready</li> </ul> </li> </ul>                          |

**LL\_RCC\_GetI2CClockFreq**

|                      |                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t LL_RCC_GetI2CClockFreq (uint32_t I2CxSource)</code>                                                                                                                                                                                                                                                                                                   |
| Function description | Return I2Cx clock frequency.                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2CxSource:</b> This parameter can be one of the following values: (*) value not defined in all devices           <ul style="list-style-type: none"> <li>- <code>LL_RCC_I2C1_CLKSOURCE</code></li> <li>- <code>LL_RCC_I2C2_CLKSOURCE</code> (*)</li> <li>- <code>LL_RCC_I2C3_CLKSOURCE</code> (*)</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>I2C:</b> clock frequency (in Hz)           <ul style="list-style-type: none"> <li>- <code>LL_RCC_PERIPH_FREQUENCY_NO</code> indicates that HSI oscillator is not ready</li> </ul> </li> </ul>                                                                                                                            |

**LL\_RCC\_GetI2SClockFreq**

|                      |                                                                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>uint32_t LL_RCC_GetI2SClockFreq (uint32_t I2SxSource)</code>                                                                                                                                                         |
| Function description | Return I2Sx clock frequency.                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>I2SxSource:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- <code>LL_RCC_I2S_CLKSOURCE</code></li> </ul> </li> </ul> |

- Return values
- **I2S:** clock frequency (in Hz)
    - LL\_RCC\_PERIPH\_FREQUENCY\_NA indicates that external clock is used

### **LL\_RCC\_GetUSBClockFreq**

- Function name      **uint32\_t LL\_RCC\_GetUSBClockFreq (uint32\_t USBxSource)**
- Function description      Return USBx clock frequency.
- Parameters
- **USBxSource:** This parameter can be one of the following values:
    - LL\_RCC\_USB\_CLKSOURCE
- Return values
- **USB:** clock frequency (in Hz)
    - LL\_RCC\_PERIPH\_FREQUENCY\_NO indicates that oscillator (HSI48) or PLL is not ready
    - LL\_RCC\_PERIPH\_FREQUENCY\_NA indicates that no clock source selected

### **LL\_RCC\_GetADCClockFreq**

- Function name      **uint32\_t LL\_RCC\_GetADCClockFreq (uint32\_t ADCxSource)**
- Function description      Return ADCx clock frequency.
- Parameters
- **ADCxSource:** This parameter can be one of the following values: (\*) value not defined in all devices
    - LL\_RCC\_ADC\_CLKSOURCE (\*)
    - LL\_RCC\_ADC1\_CLKSOURCE (\*)
    - LL\_RCC\_ADC12\_CLKSOURCE (\*)
    - LL\_RCC\_ADC34\_CLKSOURCE (\*)
- Return values
- **ADC:** clock frequency (in Hz)

### **LL\_RCC\_GetTIMClockFreq**

- Function name      **uint32\_t LL\_RCC\_GetTIMClockFreq (uint32\_t TIMxSource)**
- Function description      Return TIMx clock frequency.
- Parameters
- **TIMxSource:** This parameter can be one of the following values: (\*) value not defined in all devices
    - LL\_RCC\_TIM1\_CLKSOURCE
    - LL\_RCC\_TIM8\_CLKSOURCE (\*)
    - LL\_RCC\_TIM15\_CLKSOURCE (\*)
    - LL\_RCC\_TIM16\_CLKSOURCE (\*)
    - LL\_RCC\_TIM17\_CLKSOURCE (\*)
    - LL\_RCC\_TIM20\_CLKSOURCE (\*)
    - LL\_RCC\_TIM2\_CLKSOURCE (\*)
    - LL\_RCC\_TIM34\_CLKSOURCE (\*)
- Return values
- **TIM:** clock frequency (in Hz)

**LL\_RCC\_GetHRTIMClockFreq**

Function name      **uint32\_t LL\_RCC\_GetHRTIMClockFreq (uint32\_t HRTIMxSource)**

Function description

## 74.3 RCC Firmware driver defines

### 74.3.1 RCC

*Peripheral ADC12 clock source selection*

|                                 |                                               |
|---------------------------------|-----------------------------------------------|
| LL_RCC_ADC12_CLKSRC_HCLK        | ADC12 clock disabled, ADC12 can use AHB clock |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_1   | ADC12 PLL clock divided by 1                  |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_2   | ADC12 PLL clock divided by 2                  |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_4   | ADC12 PLL clock divided by 4                  |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_6   | ADC12 PLL clock divided by 6                  |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_8   | ADC12 PLL clock divided by 8                  |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_10  | ADC12 PLL clock divided by 10                 |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_12  | ADC12 PLL clock divided by 12                 |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_16  | ADC12 PLL clock divided by 16                 |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_32  | ADC12 PLL clock divided by 32                 |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_64  | ADC12 PLL clock divided by 64                 |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_128 | ADC12 PLL clock divided by 128                |
| LL_RCC_ADC12_CLKSRC_PLL_DIV_256 | ADC12 PLL clock divided by 256                |

*Peripheral ADC34 clock source selection*

|                                 |                                               |
|---------------------------------|-----------------------------------------------|
| LL_RCC_ADC34_CLKSRC_HCLK        | ADC34 clock disabled, ADC34 can use AHB clock |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_1   | ADC34 PLL clock divided by 1                  |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_2   | ADC34 PLL clock divided by 2                  |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_4   | ADC34 PLL clock divided by 4                  |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_6   | ADC34 PLL clock divided by 6                  |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_8   | ADC34 PLL clock divided by 8                  |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_10  | ADC34 PLL clock divided by 10                 |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_12  | ADC34 PLL clock divided by 12                 |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_16  | ADC34 PLL clock divided by 16                 |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_32  | ADC34 PLL clock divided by 32                 |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_64  | ADC34 PLL clock divided by 64                 |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_128 | ADC34 PLL clock divided by 128                |
| LL_RCC_ADC34_CLKSRC_PLL_DIV_256 | ADC34 PLL clock divided by 256                |

***Peripheral ADC get clock source***

LL\_RCC\_ADC12\_CLKSOURCE ADC12 Clock source selection

LL\_RCC\_ADC34\_CLKSOURCE ADC34 Clock source selection

***APB low-speed prescaler (APB1)***

LL\_RCC\_APB1\_DIV\_1 HCLK not divided

LL\_RCC\_APB1\_DIV\_2 HCLK divided by 2

LL\_RCC\_APB1\_DIV\_4 HCLK divided by 4

LL\_RCC\_APB1\_DIV\_8 HCLK divided by 8

LL\_RCC\_APB1\_DIV\_16 HCLK divided by 16

***APB high-speed prescaler (APB2)***

LL\_RCC\_APB2\_DIV\_1 HCLK not divided

LL\_RCC\_APB2\_DIV\_2 HCLK divided by 2

LL\_RCC\_APB2\_DIV\_4 HCLK divided by 4

LL\_RCC\_APB2\_DIV\_8 HCLK divided by 8

LL\_RCC\_APB2\_DIV\_16 HCLK divided by 16

***Clear Flags Defines***

LL\_RCC\_CIR\_LSIRDYC LSI Ready Interrupt Clear

LL\_RCC\_CIR\_LSERDYC LSE Ready Interrupt Clear

LL\_RCC\_CIR\_HSIRDYC HSI Ready Interrupt Clear

LL\_RCC\_CIR\_HSERDYC HSE Ready Interrupt Clear

LL\_RCC\_CIR\_PLLRDYC PLL Ready Interrupt Clear

LL\_RCC\_CIR\_CSSC Clock Security System Interrupt Clear

***Get Flags Defines***

LL\_RCC\_CIR\_LSIRDYF LSI Ready Interrupt flag

LL\_RCC\_CIR\_LSERDYF LSE Ready Interrupt flag

LL\_RCC\_CIR\_HSIRDYF HSI Ready Interrupt flag

LL\_RCC\_CIR\_HSERDYF HSE Ready Interrupt flag

LL\_RCC\_CFGR\_MCOF MCO flag

LL\_RCC\_CIR\_PLLRDYF PLL Ready Interrupt flag

LL\_RCC\_CIR\_CSSF Clock Security System Interrupt flag

LL\_RCC\_CSR\_OBLRSTF OBL reset flag

LL\_RCC\_CSR\_PINRSTF PIN reset flag

LL\_RCC\_CSR\_PORRSTF POR/PDR reset flag

LL\_RCC\_CSR\_SFTRSTF Software Reset flag

LL\_RCC\_CSR\_IWDGRSTF Independent Watchdog reset flag

LL\_RCC\_CSR\_WWDGRSTF Window watchdog reset flag

---

|                                    |                                 |
|------------------------------------|---------------------------------|
| <code>LL_RCC_CSR_LPWRSTF</code>    | Low-Power reset flag            |
| <code>LL_RCC_CSR_V18PWRRSTF</code> | Reset flag of the 1.8 V domain. |

***Peripheral I2C get clock source***

|                                    |                             |
|------------------------------------|-----------------------------|
| <code>LL_RCC_I2C1_CLKSOURCE</code> | I2C1 Clock source selection |
| <code>LL_RCC_I2C2_CLKSOURCE</code> | I2C2 Clock source selection |

***Peripheral I2C clock source selection***

|                                           |                                                |
|-------------------------------------------|------------------------------------------------|
| <code>LL_RCC_I2C1_CLKSOURCE_HSI</code>    | HSI oscillator clock used as I2C1 clock source |
| <code>LL_RCC_I2C1_CLKSOURCE_SYSCLK</code> | System clock selected as I2C1 clock source     |
| <code>LL_RCC_I2C2_CLKSOURCE_HSI</code>    | HSI oscillator clock used as I2C2 clock source |
| <code>LL_RCC_I2C2_CLKSOURCE_SYSCLK</code> | System clock selected as I2C2 clock source     |

***Peripheral I2S get clock source***

|                                   |                            |
|-----------------------------------|----------------------------|
| <code>LL_RCC_I2S_CLKSOURCE</code> | I2S Clock source selection |
|-----------------------------------|----------------------------|

***Peripheral I2S clock source selection***

|                                          |                                             |
|------------------------------------------|---------------------------------------------|
| <code>LL_RCC_I2S_CLKSOURCE_SYSCLK</code> | System clock selected as I2S clock source   |
| <code>LL_RCC_I2S_CLKSOURCE_PIN</code>    | External clock selected as I2S clock source |

***IT Defines***

|                                  |                            |
|----------------------------------|----------------------------|
| <code>LL_RCC_CIR_LSIRDYIE</code> | LSI Ready Interrupt Enable |
| <code>LL_RCC_CIR_LSERDYIE</code> | LSE Ready Interrupt Enable |
| <code>LL_RCC_CIR_HSIRDYIE</code> | HSI Ready Interrupt Enable |
| <code>LL_RCC_CIR_HSERDYIE</code> | HSE Ready Interrupt Enable |
| <code>LL_RCC_CIR_PLLRDYIE</code> | PLL Ready Interrupt Enable |

***LSE oscillator drive capability***

|                                         |                                          |
|-----------------------------------------|------------------------------------------|
| <code>LL_RCC_LSEDRIVE_LOW</code>        | Xtal mode lower driving capability       |
| <code>LL_RCC_LSEDRIVE_MEDIUMLOW</code>  | Xtal mode medium low driving capability  |
| <code>LL_RCC_LSEDRIVE_MEDIUMHIGH</code> | Xtal mode medium high driving capability |
| <code>LL_RCC_LSEDRIVE_HIGH</code>       | Xtal mode higher driving capability      |

***MCO1 SOURCE selection***

|                                             |                                      |
|---------------------------------------------|--------------------------------------|
| <code>LL_RCC_MCO1SOURCE_NO CLOCK</code>     | MCO output disabled, no clock on MCO |
| <code>LL_RCC_MCO1SOURCE_SYSCLK</code>       | SYSCLK selection as MCO source       |
| <code>LL_RCC_MCO1SOURCE_HSI</code>          | HSI selection as MCO source          |
| <code>LL_RCC_MCO1SOURCE_HSE</code>          | HSE selection as MCO source          |
| <code>LL_RCC_MCO1SOURCE_LSI</code>          | LSI selection as MCO source          |
| <code>LL_RCC_MCO1SOURCE_LSE</code>          | LSE selection as MCO source          |
| <code>LL_RCC_MCO1SOURCE_PLLCLK_DIV_2</code> | PLL clock divided by 2               |

***MCO1 prescaler***

|                                |                        |
|--------------------------------|------------------------|
| <code>LL_RCC_MCO1_DIV_1</code> | MCO Clock divided by 1 |
|--------------------------------|------------------------|

**Oscillator Values adaptation**

|           |                                   |
|-----------|-----------------------------------|
| HSE_VALUE | Value of the HSE oscillator in Hz |
| HSI_VALUE | Value of the HSI oscillator in Hz |
| LSE_VALUE | Value of the LSE oscillator in Hz |
| LSI_VALUE | Value of the LSI oscillator in Hz |

**Peripheral clock frequency**

|                            |                                                |
|----------------------------|------------------------------------------------|
| LL_RCC_PERIPH_FREQUENCY_NO | No clock enabled for the peripheral            |
| LL_RCC_PERIPH_FREQUENCY_NA | Frequency cannot be provided as external clock |

**PLL SOURCE**

|                             |                                                           |
|-----------------------------|-----------------------------------------------------------|
| LL_RCC_PLLSOURCE_HSE        | HSE/PREDIV clock selected as PLL entry clock source       |
| LL_RCC_PLLSOURCE_HSI_DIV_2  | HSI clock divided by 2 selected as PLL entry clock source |
| LL_RCC_PLLSOURCE_HSE_DIV_1  | HSE clock selected as PLL entry clock source              |
| LL_RCC_PLLSOURCE_HSE_DIV_2  | HSE/2 clock selected as PLL entry clock source            |
| LL_RCC_PLLSOURCE_HSE_DIV_3  | HSE/3 clock selected as PLL entry clock source            |
| LL_RCC_PLLSOURCE_HSE_DIV_4  | HSE/4 clock selected as PLL entry clock source            |
| LL_RCC_PLLSOURCE_HSE_DIV_5  | HSE/5 clock selected as PLL entry clock source            |
| LL_RCC_PLLSOURCE_HSE_DIV_6  | HSE/6 clock selected as PLL entry clock source            |
| LL_RCC_PLLSOURCE_HSE_DIV_7  | HSE/7 clock selected as PLL entry clock source            |
| LL_RCC_PLLSOURCE_HSE_DIV_8  | HSE/8 clock selected as PLL entry clock source            |
| LL_RCC_PLLSOURCE_HSE_DIV_9  | HSE/9 clock selected as PLL entry clock source            |
| LL_RCC_PLLSOURCE_HSE_DIV_10 | HSE/10 clock selected as PLL entry clock source           |
| LL_RCC_PLLSOURCE_HSE_DIV_11 | HSE/11 clock selected as PLL entry clock source           |
| LL_RCC_PLLSOURCE_HSE_DIV_12 | HSE/12 clock selected as PLL entry clock source           |
| LL_RCC_PLLSOURCE_HSE_DIV_13 | HSE/13 clock selected as PLL entry clock source           |
| LL_RCC_PLLSOURCE_HSE_DIV_14 | HSE/14 clock selected as PLL entry clock source           |
| LL_RCC_PLLSOURCE_HSE_DIV_15 | HSE/15 clock selected as PLL entry clock source           |
| LL_RCC_PLLSOURCE_HSE_DIV_16 | HSE/16 clock selected as PLL entry clock source           |

**PLL Multiplicator factor**

|                  |                   |
|------------------|-------------------|
| LL_RCC_PLL_MUL_2 | PLL input clock*2 |
| LL_RCC_PLL_MUL_3 | PLL input clock*3 |
| LL_RCC_PLL_MUL_4 | PLL input clock*4 |
| LL_RCC_PLL_MUL_5 | PLL input clock*5 |
| LL_RCC_PLL_MUL_6 | PLL input clock*6 |
| LL_RCC_PLL_MUL_7 | PLL input clock*7 |
| LL_RCC_PLL_MUL_8 | PLL input clock*8 |

---

|                   |                    |
|-------------------|--------------------|
| LL_RCC_PLL_MUL_9  | PLL input clock*9  |
| LL_RCC_PLL_MUL_10 | PLL input clock*10 |
| LL_RCC_PLL_MUL_11 | PLL input clock*11 |
| LL_RCC_PLL_MUL_12 | PLL input clock*12 |
| LL_RCC_PLL_MUL_13 | PLL input clock*13 |
| LL_RCC_PLL_MUL_14 | PLL input clock*14 |
| LL_RCC_PLL_MUL_15 | PLL input clock*15 |
| LL_RCC_PLL_MUL_16 | PLL input clock*16 |

**PREDIV Division factor**

|                      |                                  |
|----------------------|----------------------------------|
| LL_RCC_PREDIV_DIV_1  | PREDIV input clock not divided   |
| LL_RCC_PREDIV_DIV_2  | PREDIV input clock divided by 2  |
| LL_RCC_PREDIV_DIV_3  | PREDIV input clock divided by 3  |
| LL_RCC_PREDIV_DIV_4  | PREDIV input clock divided by 4  |
| LL_RCC_PREDIV_DIV_5  | PREDIV input clock divided by 5  |
| LL_RCC_PREDIV_DIV_6  | PREDIV input clock divided by 6  |
| LL_RCC_PREDIV_DIV_7  | PREDIV input clock divided by 7  |
| LL_RCC_PREDIV_DIV_8  | PREDIV input clock divided by 8  |
| LL_RCC_PREDIV_DIV_9  | PREDIV input clock divided by 9  |
| LL_RCC_PREDIV_DIV_10 | PREDIV input clock divided by 10 |
| LL_RCC_PREDIV_DIV_11 | PREDIV input clock divided by 11 |
| LL_RCC_PREDIV_DIV_12 | PREDIV input clock divided by 12 |
| LL_RCC_PREDIV_DIV_13 | PREDIV input clock divided by 13 |
| LL_RCC_PREDIV_DIV_14 | PREDIV input clock divided by 14 |
| LL_RCC_PREDIV_DIV_15 | PREDIV input clock divided by 15 |
| LL_RCC_PREDIV_DIV_16 | PREDIV input clock divided by 16 |

**RTC clock source selection**

|                                |                                                      |
|--------------------------------|------------------------------------------------------|
| LL_RCC_RTC_CLKSOURCE_NONE      | No clock used as RTC clock                           |
| LL_RCC_RTC_CLKSOURCE_LSE       | LSE oscillator clock used as RTC clock               |
| LL_RCC_RTC_CLKSOURCE_LSI       | LSI oscillator clock used as RTC clock               |
| LL_RCC_RTC_CLKSOURCE_HSE_DIV32 | HSE oscillator clock divided by 32 used as RTC clock |

**AHB prescaler**

|                      |                      |
|----------------------|----------------------|
| LL_RCC_SYSCLK_DIV_1  | SYSCLK not divided   |
| LL_RCC_SYSCLK_DIV_2  | SYSCLK divided by 2  |
| LL_RCC_SYSCLK_DIV_4  | SYSCLK divided by 4  |
| LL_RCC_SYSCLK_DIV_8  | SYSCLK divided by 8  |
| LL_RCC_SYSCLK_DIV_16 | SYSCLK divided by 16 |

LL\_RCC\_SYSCLK\_DIV\_64    SYSCLK divided by 64  
 LL\_RCC\_SYSCLK\_DIV\_128    SYSCLK divided by 128  
 LL\_RCC\_SYSCLK\_DIV\_256    SYSCLK divided by 256  
 LL\_RCC\_SYSCLK\_DIV\_512    SYSCLK divided by 512

**System clock switch**

LL\_RCC\_SYS\_CLKSOURCE\_HSI    HSI selection as system clock  
 LL\_RCC\_SYS\_CLKSOURCE\_HSE    HSE selection as system clock  
 LL\_RCC\_SYS\_CLKSOURCE\_PLL    PLL selection as system clock

**System clock switch status**

LL\_RCC\_SYS\_CLKSOURCE\_STATUS\_HSI    HSI used as system clock  
 LL\_RCC\_SYS\_CLKSOURCE\_STATUS\_HSE    HSE used as system clock  
 LL\_RCC\_SYS\_CLKSOURCE\_STATUS\_PLL    PLL used as system clock

**TIMx Peripheral TIM get clock source**

LL\_RCC\_TIM1\_CLKSOURCE    TIM1 Clock source selection  
 LL\_RCC\_TIM8\_CLKSOURCE    TIM8 Clock source selection

**Peripheral TIM clock source selection**

LL\_RCC\_TIM1\_CLKSOURCE\_PCLK2    PCLK2 used as TIM1 clock source  
 LL\_RCC\_TIM1\_CLKSOURCE\_PLL    PLL clock used as TIM1 clock source  
 LL\_RCC\_TIM8\_CLKSOURCE\_PCLK2    PCLK2 used as TIM8 clock source  
 LL\_RCC\_TIM8\_CLKSOURCE\_PLL    PLL clock used as TIM8 clock source

**Peripheral UART get clock source**

LL\_RCC\_UART4\_CLKSOURCE    UART4 Clock source selection  
 LL\_RCC\_UART5\_CLKSOURCE    UART5 Clock source selection

**Peripheral UART clock source selection**

LL\_RCC\_UART4\_CLKSOURCE\_PCLK1    PCLK1 clock used as UART4 clock source  
 LL\_RCC\_UART4\_CLKSOURCE\_SYSCLK    System clock selected as UART4 clock source  
 LL\_RCC\_UART4\_CLKSOURCE\_LSE    LSE oscillator clock used as UART4 clock source  
 LL\_RCC\_UART4\_CLKSOURCE\_HSI    HSI oscillator clock used as UART4 clock source  
 LL\_RCC\_UART5\_CLKSOURCE\_PCLK1    PCLK1 clock used as UART5 clock source  
 LL\_RCC\_UART5\_CLKSOURCE\_SYSCLK    System clock selected as UART5 clock source  
 LL\_RCC\_UART5\_CLKSOURCE\_LSE    LSE oscillator clock used as UART5 clock source  
 LL\_RCC\_UART5\_CLKSOURCE\_HSI    HSI oscillator clock used as UART5 clock source

***Peripheral USART get clock source***

`LL_RCC_USART1_CLKSOURCE` USART1 Clock source selection

`LL_RCC_USART2_CLKSOURCE` USART2 Clock source selection

`LL_RCC_USART3_CLKSOURCE` USART3 Clock source selection

***Peripheral USART clock source selection***

`LL_RCC_USART1_CLKSOURCE_PCLK2` PCLK2 clock used as USART1 clock source

`LL_RCC_USART1_CLKSOURCE_SYSCLK` System clock selected as USART1 clock source

`LL_RCC_USART1_CLKSOURCE_LSE` LSE oscillator clock used as USART1 clock source

`LL_RCC_USART1_CLKSOURCE_HSI` HSI oscillator clock used as USART1 clock source

`LL_RCC_USART2_CLKSOURCE_PCLK1` PCLK1 clock used as USART2 clock source

`LL_RCC_USART2_CLKSOURCE_SYSCLK` System clock selected as USART2 clock source

`LL_RCC_USART2_CLKSOURCE_LSE` LSE oscillator clock used as USART2 clock source

`LL_RCC_USART2_CLKSOURCE_HSI` HSI oscillator clock used as USART2 clock source

`LL_RCC_USART3_CLKSOURCE_PCLK1` PCLK1 clock used as USART3 clock source

`LL_RCC_USART3_CLKSOURCE_SYSCLK` System clock selected as USART3 clock source

`LL_RCC_USART3_CLKSOURCE_LSE` LSE oscillator clock used as USART3 clock source

`LL_RCC_USART3_CLKSOURCE_HSI` HSI oscillator clock used as USART3 clock source

***Peripheral USB get clock source***

`LL_RCC_USB_CLKSOURCE` USB Clock source selection

***Peripheral USB clock source selection***

`LL_RCC_USB_CLKSOURCE_PLL` USB prescaler is PLL clock divided by 1

`LL_RCC_USB_CLKSOURCE_PLL_DIV_1_5` USB prescaler is PLL clock divided by 1.5

***Calculate frequencies*****\_\_LL\_RCC\_CALC\_PLLCL** **Description:**

`K_FREQ`

- Helper macro to calculate the PLLCLK frequency.

**Parameters:**

- `__INPUTFREQ__`: PLL Input frequency (based on HSE div Prediv / HSI div 2)
- `__PLLMUL__`: This parameter can be one of the following values:
  - `LL_RCC_PLL_MUL_2`
  - `LL_RCC_PLL_MUL_3`
  - `LL_RCC_PLL_MUL_4`

- LL\_RCC\_PLL\_MUL\_5
- LL\_RCC\_PLL\_MUL\_6
- LL\_RCC\_PLL\_MUL\_7
- LL\_RCC\_PLL\_MUL\_8
- LL\_RCC\_PLL\_MUL\_9
- LL\_RCC\_PLL\_MUL\_10
- LL\_RCC\_PLL\_MUL\_11
- LL\_RCC\_PLL\_MUL\_12
- LL\_RCC\_PLL\_MUL\_13
- LL\_RCC\_PLL\_MUL\_14
- LL\_RCC\_PLL\_MUL\_15
- LL\_RCC\_PLL\_MUL\_16

**Return value:**

- PLL: clock frequency (in Hz)

**Notes:**

- ex: \_\_LL\_RCC\_CALC\_PLLCLK\_FREQ (HSE\_VALUE / (LL\_RCC\_PLL\_GetPrediv () + 1), LL\_RCC\_PLL\_GetMultiplicator());

[\\_\\_LL\\_RCC\\_CALC\\_HCLK\\_FREQ](#)**Description:**

- Helper macro to calculate the HCLK frequency.

**Parameters:**

- \_\_SYSCLKFREQ\_\_: SYSCLK frequency (based on HSE/HSI/PLLCLK)
- \_\_AHBPRESCALER\_\_: This parameter can be one of the following values:
  - LL\_RCC\_SYSCLK\_DIV\_1
  - LL\_RCC\_SYSCLK\_DIV\_2
  - LL\_RCC\_SYSCLK\_DIV\_4
  - LL\_RCC\_SYSCLK\_DIV\_8
  - LL\_RCC\_SYSCLK\_DIV\_16
  - LL\_RCC\_SYSCLK\_DIV\_64
  - LL\_RCC\_SYSCLK\_DIV\_128
  - LL\_RCC\_SYSCLK\_DIV\_256
  - LL\_RCC\_SYSCLK\_DIV\_512

**Return value:**

- HCLK: clock frequency (in Hz)

**Notes:**

- : \_\_AHBPRESCALER\_\_ be retrieved by LL\_RCC\_GetAHBPrescaler ex: \_\_LL\_RCC\_CALC\_HCLK\_FREQ(LL\_RCC\_GetAHBPrescaler())

[\\_\\_LL\\_RCC\\_CALC\\_PCLK1\\_FREQ](#)**Description:**

- Helper macro to calculate the PCLK1 frequency (ABP1)

**Parameters:**

- `__HCLKFREQ__`: HCLK frequency
- `__APB1PRESCALER__`: This parameter can be one of the following values:
  - `LL_RCC_APB1_DIV_1`
  - `LL_RCC_APB1_DIV_2`
  - `LL_RCC_APB1_DIV_4`
  - `LL_RCC_APB1_DIV_8`
  - `LL_RCC_APB1_DIV_16`

**Return value:**

- PCLK1: clock frequency (in Hz)

**Notes:**

- : `__APB1PRESCALER__` be retrieved by `LL_RCC_GetAPB1Prescaler` ex:  
`__LL_RCC_CALC_PCLK1_FREQ(LL_RCC_GetAPB1P  
rescaler())`

`__LL_RCC_CALC_PCLK2  
_FREQ`**Description:**

- Helper macro to calculate the PCLK2 frequency (ABP2)

**Parameters:**

- `__HCLKFREQ__`: HCLK frequency
- `__APB2PRESCALER__`: This parameter can be one of the following values:
  - `LL_RCC_APB2_DIV_1`
  - `LL_RCC_APB2_DIV_2`
  - `LL_RCC_APB2_DIV_4`
  - `LL_RCC_APB2_DIV_8`
  - `LL_RCC_APB2_DIV_16`

**Return value:**

- PCLK2: clock frequency (in Hz)

**Notes:**

- : `__APB2PRESCALER__` be retrieved by `LL_RCC_GetAPB2Prescaler` ex:  
`__LL_RCC_CALC_PCLK2_FREQ(LL_RCC_GetAPB2P  
rescaler())`

***Common Write and read registers Macros***`LL_RCC_WriteReg`**Description:**

- Write a value in RCC register.

**Parameters:**

- `__REG__`: Register to be written
- `__VALUE__`: Value to be written in the register

**Return value:**

- None

LL\_RCC\_ReadReg

**Description:**

- Read a value in RCC register.

**Parameters:**

- \_\_REG\_\_: Register to be read

**Return value:**

- Register: value

## 75 LL RTC Generic Driver

### 75.1 RTC Firmware driver registers structures

#### 75.1.1 LL\_RTC\_InitTypeDef

##### Data Fields

- *uint32\_t HourFormat*
- *uint32\_t AsynchPrescaler*
- *uint32\_t SynchPrescaler*

##### Field Documentation

- ***uint32\_t LL\_RTC\_InitTypeDef::HourFormat***  
Specifies the RTC Hours Format. This parameter can be a value of **RTC\_LL\_EC\_HOURFORMAT**This feature can be modified afterwards using unitary function **LL\_RTC\_SetHourFormat()**.
- ***uint32\_t LL\_RTC\_InitTypeDef::AsynchPrescaler***  
Specifies the RTC Asynchronous Predivider value. This parameter must be a number between Min\_Data = 0x00 and Max\_Data = 0x7FThis feature can be modified afterwards using unitary function **LL\_RTC\_SetAsynchPrescaler()**.
- ***uint32\_t LL\_RTC\_InitTypeDef::SynchPrescaler***  
Specifies the RTC Synchronous Predivider value. This parameter must be a number between Min\_Data = 0x00 and Max\_Data = 0x7FFFThis feature can be modified afterwards using unitary function **LL\_RTC\_SetSynchPrescaler()**.

#### 75.1.2 LL\_RTC\_TimeTypeDef

##### Data Fields

- *uint32\_t TimeFormat*
- *uint8\_t Hours*
- *uint8\_t Minutes*
- *uint8\_t Seconds*

##### Field Documentation

- ***uint32\_t LL\_RTC\_TimeTypeDef::TimeFormat***  
Specifies the RTC AM/PM Time. This parameter can be a value of **RTC\_LL\_EC\_TIME\_FORMAT**This feature can be modified afterwards using unitary function **LL\_RTC\_TIME\_SetFormat()**.
- ***uint8\_t LL\_RTC\_TimeTypeDef::Hours***  
Specifies the RTC Time Hours. This parameter must be a number between Min\_Data = 0 and Max\_Data = 12 if the **LL\_RTC\_TIME\_FORMAT\_PM** is selected. This parameter must be a number between Min\_Data = 0 and Max\_Data = 23 if the **LL\_RTC\_TIME\_FORMAT\_AM\_OR\_24** is selected.This feature can be modified afterwards using unitary function **LL\_RTC\_TIME\_SetHour()**.
- ***uint8\_t LL\_RTC\_TimeTypeDef::Minutes***  
Specifies the RTC Time Minutes. This parameter must be a number between Min\_Data = 0 and Max\_Data = 59This feature can be modified afterwards using unitary function **LL\_RTC\_TIME\_SetMinute()**.
- ***uint8\_t LL\_RTC\_TimeTypeDef::Seconds***  
Specifies the RTC Time Seconds. This parameter must be a number between

Min\_Data = 0 and Max\_Data = 59This feature can be modified afterwards using unitary function **LL\_RTC\_TIME\_SetSecond()**.

### 75.1.3 LL\_RTC\_DateTypeDef

#### Data Fields

- *uint8\_t WeekDay*
- *uint8\_t Month*
- *uint8\_t Day*
- *uint8\_t Year*

#### Field Documentation

- ***uint8\_t LL\_RTC\_DateTypeDef::WeekDay***  
Specifies the RTC Date WeekDay. This parameter can be a value of **RTC\_LL\_EC\_WEEKDAY**This feature can be modified afterwards using unitary function **LL\_RTC\_DATE\_SetWeekDay()**.
- ***uint8\_t LL\_RTC\_DateTypeDef::Month***  
Specifies the RTC Date Month. This parameter can be a value of **RTC\_LL\_EC\_MONTH**This feature can be modified afterwards using unitary function **LL\_RTC\_DATE\_SetMonth()**.
- ***uint8\_t LL\_RTC\_DateTypeDef::Day***  
Specifies the RTC Date Day. This parameter must be a number between Min\_Data = 1 and Max\_Data = 31This feature can be modified afterwards using unitary function **LL\_RTC\_DATE\_SetDay()**.
- ***uint8\_t LL\_RTC\_DateTypeDef::Year***  
Specifies the RTC Date Year. This parameter must be a number between Min\_Data = 0 and Max\_Data = 99This feature can be modified afterwards using unitary function **LL\_RTC\_DATE\_SetYear()**.

### 75.1.4 LL\_RTC\_AlarmTypeDef

#### Data Fields

- ***LL\_RTC\_TimeTypeDef AlarmTime***
- ***uint32\_t AlarmMask***
- ***uint32\_t AlarmDateWeekDaySel***
- ***uint8\_t AlarmDateWeekDay***

#### Field Documentation

- ***LL\_RTC\_TimeTypeDef LL\_RTC\_AlarmTypeDef::AlarmTime***  
Specifies the RTC Alarm Time members.
- ***uint32\_t LL\_RTC\_AlarmTypeDef::AlarmMask***  
Specifies the RTC Alarm Masks. This parameter can be a value of **RTC\_LL\_EC\_ALMA\_MASK** for ALARM A or **RTC\_LL\_EC\_ALMB\_MASK** for ALARM B.This feature can be modified afterwards using unitary function **LL\_RTC\_ALMA\_SetMask()** for ALARM A or **LL\_RTC\_ALMB\_SetMask()** for ALARM B
- ***uint32\_t LL\_RTC\_AlarmTypeDef::AlarmDateWeekDaySel***  
Specifies the RTC Alarm is on day or WeekDay. This parameter can be a value of **RTC\_LL\_EC\_ALMA\_WEEKDAY\_SELECTION** for ALARM A or **RTC\_LL\_EC\_ALMB\_WEEKDAY\_SELECTION** for ALARM BThis feature can be modified afterwards using unitary function **LL\_RTC\_ALMA\_EnableWeekday()** or **LL\_RTC\_ALMA\_DisableWeekday()** for ALARM A or **LL\_RTC\_ALMB\_EnableWeekday()** or **LL\_RTC\_ALMB\_DisableWeekday()** for ALARM B

- ***uint8\_t LL\_RTC\_AlarmTypeDef::AlarmDateWeekDay***

Specifies the RTC Alarm Day/WeekDay. If AlarmDateWeekDaySel set to day, this parameter must be a number between Min\_Data = 1 and Max\_Data = 31. This feature can be modified afterwards using unitary function **LL\_RTC\_ALMA\_SetDay()** for ALARM A or **LL\_RTC\_ALMB\_SetDay()** for ALARM B. If AlarmDateWeekDaySel set to Weekday, this parameter can be a value of **RTC\_LL\_EC\_WEEKDAY**. This feature can be modified afterwards using unitary function **LL\_RTC\_ALMA\_SetWeekDay()** for ALARM A or **LL\_RTC\_ALMB\_SetWeekDay()** for ALARM B.

## 75.2 RTC Firmware driver API description

### 75.2.1 Detailed description of functions

#### LL\_RTC\_SetHourFormat

|                                                   |                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_RTC_SetHourFormat<br/>(RTC_TypeDef * RTCx, uint32_t HourFormat)</code></b>                                                                                                                                                                     |
| Function description                              | Set Hours format (24 hour/day or AM/PM hour format)                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>HourFormat:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_RTC_HOURFORMAT_24HOUR</li> <li>– LL_RTC_HOURFORMAT_AMPM</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. <b>LL_RTC_DisableWriteProtection</b> function should be called before.</li> <li>• It can be written in initialization mode only (<b>LL_RTC_EnableInitMode</b> function)</li> </ul>                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR FMT LL_RTC_SetHourFormat</li> </ul>                                                                                                                                                                                                |

#### LL\_RTC\_GetHourFormat

|                                                   |                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_RTC_GetHourFormat<br/>(RTC_TypeDef * RTCx)</code></b>                                                                                                                                      |
| Function description                              | Get Hours format (24 hour/day or AM/PM hour format)                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_RTC_HOURFORMAT_24HOUR</li> <li>– LL_RTC_HOURFORMAT_AMPM</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR FMT LL_RTC_GetHourFormat</li> </ul>                                                                                                                                                |

**LL\_RTC\_SetAlarmOutEvent**

|                                                   |                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_SetAlarmOutEvent<br/>(RTC_TypeDef * RTCx, uint32_t AlarmOutput)</code>                                                                                                                                                                                                                                       |
| Function description                              | Select the flag to be routed to RTC_ALARM output.                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>AlarmOutput:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_RTC_ALARMOUT_DISABLE</li> <li>– LL_RTC_ALARMOUT_ALMA</li> <li>– LL_RTC_ALARMOUT_ALMB</li> <li>– LL_RTC_ALARMOUT_WAKEUP</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul>                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR OSEL LL_RTC_SetAlarmOutEvent</li> </ul>                                                                                                                                                                                                                                                            |

**LL\_RTC\_GetAlarmOutEvent**

|                                                   |                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_GetAlarmOutEvent<br/>(RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                        |
| Function description                              | Get the flag to be routed to RTC_ALARM output.                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_RTC_ALARMOUT_DISABLE</li> <li>– LL_RTC_ALARMOUT_ALMA</li> <li>– LL_RTC_ALARMOUT_ALMB</li> <li>– LL_RTC_ALARMOUT_WAKEUP</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR OSEL LL_RTC_GetAlarmOutEvent</li> </ul>                                                                                                                                                                                                           |

**LL\_RTC\_SetAlarmOutputType**

|                      |                                                                                                                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_SetAlarmOutputType<br/>(RTC_TypeDef * RTCx, uint32_t Output)</code>                                                                                                                                                                                          |
| Function description | Set RTC_ALARM output type (ALARM in push-pull or open-drain output)                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Output:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_RTC_ALARM_OUTPUTTYPE_OPENDRAIN</li> <li>– LL_RTC_ALARM_OUTPUTTYPE_PUSH_PULL</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                |
| Notes                | <ul style="list-style-type: none"> <li>• Used only when RTC_ALARM is mapped on PC13</li> <li>• If all RTC alternate functions are disabled and PC13MODE =</li> </ul>                                                                                                                           |

- Reference Manual to  
LL API cross  
reference:
- TAFCR ALARMOUTTYPE LL\_RTC\_SetAlarmOutputType

### **LL\_RTC\_GetAlarmOutputType**

|                                                   |                                                                                                                                                                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RTC_GetAlarmOutputType(<br/>(RTC_TypeDef * RTCx)</code></b>                                                                                                                                                             |
| Function description                              | Get RTC_ALARM output type (ALARM in push-pull or open-drain output)                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_ALARM_OUTPUTTYPE_OPENDRAIN</li> <li>– LL_RTC_ALARM_OUTPUTTYPE_PUSH_PULL</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• used only when RTC_ALARM is mapped on PC13</li> <li>• If all RTC alternate functions are disabled and PC13MODE = 1, PC13VALUE configures the PC13 output data</li> </ul>                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TAFCR ALARMOUTTYPE LL_RTC_GetAlarmOutputType</li> </ul>                                                                                                                                                             |

### **LL\_RTC\_EnablePushPullMode**

|                                                   |                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RTC_EnablePushPullMode(<br/>(RTC_TypeDef * RTCx, uint32_t PinMask)</code></b>                                                                                                                                                                                             |
| Function description                              | Enable push-pull output on PC13, PC14 and/or PC15.                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>PinMask:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_PIN_PC13</li> <li>– LL_RTC_PIN_PC14</li> <li>– LL_RTC_PIN_PC15</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• PC13 forced to push-pull output if all RTC alternate functions are disabled</li> <li>• PC14 and PC15 forced to push-pull output if LSE is disabled</li> </ul>                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TAFCR PC13MODE LL_RTC_EnablePushPullMode</li> <li>• TAFCR PC14MODE LL_RTC_EnablePushPullMode</li> <li>• TAFCR PC15MODE LL_RTC_EnablePushPullMode</li> </ul>                                                                                                       |

**LL\_RTC\_DisablePushPullMode**

|                                                   |                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_RTC_DisablePushPullMode<br/>(RTC_TypeDef * RTCx, uint32_t PinMask)</b>                                                                                                                                                                                                 |
| Function description                              | Disable push-pull output on PC13, PC14 and/or PC15.                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>PinMask:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_RTC_PIN_PC13</li> <li>– LL_RTC_PIN_PC14</li> <li>– LL_RTC_PIN_PC15</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• PC13, PC14 and/or PC15 are controlled by the GPIO configuration registers. Consequently PC13, PC14 and/or PC15 are floating in Standby mode.</li> </ul>                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TAFCR PC13MODE LL_RTC_DisablePushPullMode</li> <li>• TAFCR PC14MODE LL_RTC_DisablePushPullMode</li> <li>• TAFCR PC15MODE LL_RTC_DisablePushPullMode</li> </ul>                                                                                          |

**LL\_RTC\_SetOutputPin**

|                                                   |                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_RTC_SetOutputPin (RTC_TypeDef<br/>* RTCx, uint32_t PinMask)</b>                                                                                                                                                                             |
| Function description                              | Set PC14 and/or PC15 to high level.                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>PinMask:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_RTC_PIN_PC14</li> <li>– LL_RTC_PIN_PC15</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• Output data configuration is possible if the LSE is disabled and PushPull output is enabled (through LL_RTC_EnablePushPullMode)</li> </ul>                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TAFCR PC14VALUE LL_RTC_SetOutputPin</li> <li>• TAFCR PC15VALUE LL_RTC_SetOutputPin</li> </ul>                                                                                                                                |

**LL\_RTC\_ResetOutputPin**

|                      |                                                                                                                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE void LL_RTC_ResetOutputPin<br/>(RTC_TypeDef * RTCx, uint32_t PinMask)</b>                                                                                                                                                                           |
| Function description | Set PC14 and/or PC15 to low level.                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>PinMask:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_RTC_PIN_PC14</li> <li>– LL_RTC_PIN_PC15</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>Output data configuration is possible if the LSE is disabled and PushPull output is enabled (through LL_RTC_EnablePushPullMode)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TAFCR PC14VALUE LL_RTC_ResetOutputPin</li> <li>TAFCR PC15VALUE LL_RTC_ResetOutputPin</li> </ul>                                            |

### LL\_RTC\_EnableInitMode

|                                                   |                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RTC_EnableInitMode(RTC_TypeDef * RTCx)</code></b>                                                                                                                                                                    |
| Function description                              | Enable initialization mode.                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>Initialization mode is used to program time and date register (RTC_TR and RTC_DR) and prescaler register (RTC_PRER). Counters are stopped and start counting from the new value when INIT is reset.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR INIT LL_RTC_EnableInitMode</li> </ul>                                                                                                                                                                      |

### LL\_RTC\_DisableInitMode

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RTC_DisableInitMode(RTC_TypeDef * RTCx)</code></b> |
| Function description                              | Disable initialization mode (Free running mode)                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR INIT LL_RTC_DisableInitMode</li> </ul>   |

### LL\_RTC\_SetOutputPolarity

|                      |                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_RTC_SetOutputPolarity(RTC_TypeDef * RTCx, uint32_t Polarity)</code></b>                                                                                                                                                                          |
| Function description | Set Output polarity (pin is low when ALRAF/ALRBF/WUTF is asserted)                                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Polarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_RTC_OUTPUTPOLARITY_PIN_HIGH</li> <li>LL_RTC_OUTPUTPOLARITY_PIN_LOW</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                     |
| Notes                | Bit is write-protected. LL_RTC_DisableWriteProtection                                                                                                                                                                                                                             |

function should be called before.

Reference Manual to  
LL API cross  
reference:

- CR POL LL\_RTC\_SetOutputPolarity

### **LL\_RTC\_GetOutputPolarity**

Function name **`_STATIC_INLINE uint32_t LL_RTC_GetOutputPolarity(RTC_TypeDef * RTCx)`**

Function description Get Output polarity.

Parameters

- **RTCx:** RTC Instance

Return values

- **Returned:** value can be one of the following values:
  - LL\_RTC\_OUTPUTPOLARITY\_PIN\_HIGH
  - LL\_RTC\_OUTPUTPOLARITY\_PIN\_LOW

Reference Manual to  
LL API cross  
reference:

- CR POL LL\_RTC\_GetOutputPolarity

### **LL\_RTC\_EnableShadowRegBypass**

Function name **`_STATIC_INLINE void LL_RTC_EnableShadowRegBypass(RTC_TypeDef * RTCx)`**

Function description Enable Bypass the shadow registers.

Parameters

- **RTCx:** RTC Instance

Return values

- **None**

Notes

- Bit is write-protected. LL\_RTC\_DisableWriteProtection function should be called before.

Reference Manual to  
LL API cross  
reference:

- CR BYPSHAD LL\_RTC\_EnableShadowRegBypass

### **LL\_RTC\_DisableShadowRegBypass**

Function name **`_STATIC_INLINE void LL_RTC_DisableShadowRegBypass(RTC_TypeDef * RTCx)`**

Function description Disable Bypass the shadow registers.

Parameters

- **RTCx:** RTC Instance

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CR BYPSHAD LL\_RTC\_DisableShadowRegBypass

### **LL\_RTC\_IsShadowRegBypassEnabled**

Function name **`_STATIC_INLINE uint32_t LL_RTC_IsShadowRegBypassEnabled (RTC_TypeDef * RTCx)`**

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function description                              | Check if Shadow registers bypass is enabled or not.                              |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | CR BYPSHAD LL_RTC_IsShadowRegBypassEnabled                                       |

### LL\_RTC\_EnableRefClock

|                                                   |                                                                                                                                                                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_RTC_EnableRefClock<br/>(RTC_TypeDef * RTCx)</code></b>                                                                                                                                            |
| Function description                              | Enable RTC_REFIN reference clock detection (50 or 60 Hz)                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> <li>It can be written in initialization mode only (LL_RTC_EnableInitMode function)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | CR REFCKON LL_RTC_EnableRefClock                                                                                                                                                                                                  |

### LL\_RTC\_DisableRefClock

|                                                   |                                                                                                                                                                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_RTC_DisableRefClock<br/>(RTC_TypeDef * RTCx)</code></b>                                                                                                                                           |
| Function description                              | Disable RTC_REFIN reference clock detection (50 or 60 Hz)                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> <li>It can be written in initialization mode only (LL_RTC_EnableInitMode function)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | CR REFCKON LL_RTC_DisableRefClock                                                                                                                                                                                                 |

### LL\_RTC\_SetAsynchPrescaler

|                      |                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_RTC_SetAsynchPrescaler<br/>(RTC_TypeDef * RTCx, uint32_t AsynchPrescaler)</code></b>                                        |
| Function description | Set Asynchronous prescaler factor.                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>AsynchPrescaler:</b> Value between Min_Data = 0 and Max_Data = 0x7F</li> </ul> |

---

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• PRER PREDIV_A LL_RTC_SetSynchPrescaler</li> </ul> |

### LL\_RTC\_SetSynchPrescaler

|                                                   |                                                                                                                                                                  |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_RTC_SetSynchPrescaler<br/>(RTC_TypeDef * RTCx, uint32_t SynchPrescaler)</code></b>                                               |
| Function description                              | Set Synchronous prescaler factor.                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>SynchPrescaler:</b> Value between Min_Data = 0 and Max_Data = 0x7FFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• PRER PREDIV_S LL_RTC_SetSynchPrescaler</li> </ul>                                                                       |

### LL\_RTC\_GetAsynchPrescaler

|                                                   |                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_RTC_GetAsynchPrescaler<br/>(RTC_TypeDef * RTCx)</code></b>             |
| Function description                              | Get Asynchronous prescaler factor.                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data = 0 and Max_Data = 0x7F</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• PRER PREDIV_A LL_RTC_GetAsynchPrescaler</li> </ul>                |

### LL\_RTC\_GetSynchPrescaler

|                                                   |                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_RTC_GetSynchPrescaler<br/>(RTC_TypeDef * RTCx)</code></b>                |
| Function description                              | Get Synchronous prescaler factor.                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data = 0 and Max_Data = 0x7FFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• PRER PREDIV_S LL_RTC_GetSynchPrescaler</li> </ul>                   |

### LL\_RTC\_EnableWriteProtection

|                      |                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_RTC_EnableWriteProtection<br/>(RTC_TypeDef * RTCx)</code></b> |
| Function description | Enable the write protection for RTC registers.                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                 |

---

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>WPR KEY LL_RTC_EnableWriteProtection</li> </ul> |

### LL\_RTC\_DisableWriteProtection

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_DisableWriteProtection(<br/>    RTC_TypeDef * RTCx)</code> |
| Function description                              | Disable the write protection for RTC registers.                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                  |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>WPR KEY LL_RTC_DisableWriteProtection</li> </ul>      |

### LL\_RTC\_TIME\_SetFormat

|                                                   |                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_TIME_SetFormat(<br/>    RTC_TypeDef * RTCx, uint32_t TimeFormat)</code>                                                                                                                                                                              |
| Function description                              | Set time format (AM/24-hour or PM notation)                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>TimeFormat:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_TIME_FORMAT_AM_OR_24</li> <li>– LL_RTC_TIME_FORMAT_PM</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> <li>It can be written in initialization mode only (LL_RTC_EnableInitMode function)</li> </ul>                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TR PM LL_RTC_TIME_SetFormat</li> </ul>                                                                                                                                                                                                          |

### LL\_RTC\_TIME\_GetFormat

|                      |                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_RTC_TIME_GetFormat(<br/>    RTC_TypeDef * RTCx)</code>                                                                                                                                                 |
| Function description | Get time format (AM or PM notation)                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                              |
| Return values        | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_TIME_FORMAT_AM_OR_24</li> <li>– LL_RTC_TIME_FORMAT_PM</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>if shadow mode is disabled (BYPSHAD=0), need to check if RSF flag is set before reading this bit</li> <li>Read either RTC_SSR or RTC_TR locks the values in the</li> </ul>                        |

Reference Manual to  
LL API cross  
reference:

higher-order calendar shadow registers until RTC\_DR is read  
(LL\_RTC\_ReadReg(RTC, DR)).

- TR PM LL\_RTC\_TIME\_SetHour

### LL\_RTC\_TIME\_SetHour

|                                                   |                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_TIME_SetHour (RTC_TypeDef * RTCx, uint32_t Hours)</code>                                                                                                                                                                                                                                                     |
| Function description                              | Set Hours in BCD format.                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Hours:</b> Value between Min_Data=0x01 and Max_Data=0x12 or between Min_Data=0x00 and Max_Data=0x23</li> </ul>                                                                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> <li>• It can be written in initialization mode only (LL_RTC_EnableInitMode function)</li> <li>• helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert hour from binary to BCD format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TR HT LL_RTC_TIME_SetHour</li> <li>• TR HU LL_RTC_TIME_SetHour</li> </ul>                                                                                                                                                                                                                            |

### LL\_RTC\_TIME\_GetHour

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_RTC_TIME_GetHour (RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                                                                        |
| Function description                              | Get Hours in BCD format.                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x01 and Max_Data=0x12 or between Min_Data=0x00 and Max_Data=0x23</li> </ul>                                                                                                                                                                                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>• if shadow mode is disabled (BYPSHAD=0), need to check if RSF flag is set before reading this bit</li> <li>• Read either RTC_SSR or RTC_TR locks the values in the higher-order calendar shadow registers until RTC_DR is read (LL_RTC_ReadReg(RTC, DR)).</li> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert hour from BCD to Binary format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TR HT LL_RTC_TIME_GetHour</li> <li>• TR HU LL_RTC_TIME_GetHour</li> </ul>                                                                                                                                                                                                                                                                                                   |

**LL\_RTC\_TIME\_SetMinute**

|                                                   |                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_TIME_SetMinute<br/>(RTC_TypeDef * RTCx, uint32_t Minutes)</code>                                                                                                                                                                                                                                               |
| Function description                              | Set Minutes in BCD format.                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Minutes:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> </ul>                                                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> <li>• It can be written in initialization mode only (LL_RTC_EnableInitMode function)</li> <li>• helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Minutes from binary to BCD format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TR MNT LL_RTC_TIME_SetMinute</li> <li>• TR MNU LL_RTC_TIME_SetMinute</li> </ul>                                                                                                                                                                                                                         |

**LL\_RTC\_TIME\_GetMinute**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_TIME_GetMinute<br/>(RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                                                                   |
| Function description                              | Get Minutes in BCD format.                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x59</li> </ul>                                                                                                                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• if shadow mode is disabled (BYPSHAD=0), need to check if RSF flag is set before reading this bit</li> <li>• Read either RTC_SSR or RTC_TR locks the values in the higher-order calendar shadow registers until RTC_DR is read (LL_RTC_ReadReg(RTC, DR)).</li> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert minute from BCD to Binary format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TR MNT LL_RTC_TIME_GetMinute</li> <li>• TR MNU LL_RTC_TIME_GetMinute</li> </ul>                                                                                                                                                                                                                                                                                               |

**LL\_RTC\_TIME\_SetSecond**

|                      |                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_TIME_SetSecond<br/>(RTC_TypeDef * RTCx, uint32_t Seconds)</code>                                                     |
| Function description | Set Seconds in BCD format.                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Seconds:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection</li> </ul>                                              |

- function should be called before.
  - It can be written in initialization mode only (LL\_RTC\_EnableInitMode function)
  - helper macro \_\_LL\_RTC\_CONVERT\_BIN2BCD is available to convert Seconds from binary to BCD format
- Reference Manual to LL API cross reference:
- TR ST LL\_RTC\_TIME\_SetSecond
  - TR SU LL\_RTC\_TIME\_SetSecond

### LL\_RTC\_TIME\_GetSecond

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_RTC_TIME_GetSecond (RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                                                                        |
| Function description                        | Get Seconds in BCD format.                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                           |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x59</li> </ul>                                                                                                                                                                                                                                                                                                               |
| Notes                                       | <ul style="list-style-type: none"> <li>• if shadow mode is disabled (BYPSHAD=0), need to check if RSF flag is set before reading this bit</li> <li>• Read either RTC_SSR or RTC_TR locks the values in the higher-order calendar shadow registers until RTC_DR is read (LL_RTC_ReadReg(RTC, DR)).</li> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Seconds from BCD to Binary format</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• TR ST LL_RTC_TIME_GetSecond</li> <li>• TR SU LL_RTC_TIME_GetSecond</li> </ul>                                                                                                                                                                                                                                                                                                  |

### LL\_RTC\_TIME\_Config

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_TIME_Config (RTC_TypeDef * RTCx, uint32_t Format12_24, uint32_t Hours, uint32_t Minutes, uint32_t Seconds)</code>                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Set time (hour, minute and second) in BCD format.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Format12_24:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_RTC_TIME_FORMAT_AM_OR_24</li> <li>- LL_RTC_TIME_FORMAT_PM</li> </ul> </li> <li>• <b>Hours:</b> Value between Min_Data=0x01 and Max_Data=0x12 or between Min_Data=0x00 and Max_Data=0x23</li> <li>• <b>Minutes:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> <li>• <b>Seconds:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> <li>• It can be written in initialization mode only (LL_RTC_EnableInitMode function)</li> </ul>                                                                                                                                                                                                                                                                                                                 |

- Reference Manual to LL API cross reference:
- TimeFormat and Hours should follow the same format
  - TR PM LL\_RTC\_TIME\_Config
  - TR HT LL\_RTC\_TIME\_Config
  - TR HU LL\_RTC\_TIME\_Config
  - TR MNT LL\_RTC\_TIME\_Config
  - TR MNU LL\_RTC\_TIME\_Config
  - TR ST LL\_RTC\_TIME\_Config
  - TR SU LL\_RTC\_TIME\_Config

### **LL\_RTC\_TIME\_Get**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE uint32_t LL_RTC_TIME_Get (RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description                        | Get time (hour, minute and second) in BCD format.                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                               |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Combination:</b> of hours, minutes and seconds (Format: 0x00HHMMSS).</li> </ul>                                                                                                                                                                                                                                                                                                                                                                 |
| Notes                                       | <ul style="list-style-type: none"> <li>• if shadow mode is disabled (BYPSHAD=0), need to check if RSF flag is set before reading this bit</li> <li>• Read either RTC_SSR or RTC_TR locks the values in the higher-order calendar shadow registers until RTC_DR is read (LL_RTC_ReadReg(RTC, DR)).</li> <li>• helper macros <code>_LL_RTC_GET_HOUR</code>, <code>_LL_RTC_GET_MINUTE</code> and <code>_LL_RTC_GET_SECOND</code> are available to get independently each parameter.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• TR HT LL_RTC_TIME_Get</li> <li>• TR HU LL_RTC_TIME_Get</li> <li>• TR MNT LL_RTC_TIME_Get</li> <li>• TR MNU LL_RTC_TIME_Get</li> <li>• TR ST LL_RTC_TIME_Get</li> <li>• TR SU LL_RTC_TIME_Get</li> </ul>                                                                                                                                                                                                                                            |

### **LL\_RTC\_TIME\_EnableDayLightStore**

|                                             |                                                                                                                                             |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE void LL_RTC_TIME_EnableDayLightStore (RTC_TypeDef * RTCx)</code>                                                       |
| Function description                        | Memorize whether the daylight saving time change has been performed.                                                                        |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                       | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR BKP LL_RTC_TIME_EnableDayLightStore</li> </ul>                                                  |

**LL\_RTC\_TIME\_DisableDayLightStore**

|                                                   |                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_TIME_DisableDayLightStore (RTC_TypeDef * RTCx)</code>                                                     |
| Function description                              | Disable memorization whether the daylight saving time change has been performed.                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR BKP LL_RTC_TIME_DisableDayLightStore</li> </ul>                                                 |

**LL\_RTC\_TIME\_IsDayLightStoreEnabled**

|                                                   |                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_TIME_IsDayLightStoreEnabled (RTC_TypeDef * RTCx)</code> |
| Function description                              | Check if RTC Day Light Saving stored operation has been enabled or not.                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR BKP LL_RTC_TIME_IsDayLightStoreEnabled</li> </ul> |

**LL\_RTC\_TIME\_DecHour**

|                                                   |                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_TIME_DecHour (RTC_TypeDef * RTCx)</code>                                                                  |
| Function description                              | Subtract 1 hour (winter time change)                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR SUB1H LL_RTC_TIME_DecHour</li> </ul>                                                            |

**LL\_RTC\_TIME\_IncHour**

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_TIME_IncHour (RTC_TypeDef * RTCx)</code>    |
| Function description | Add 1 hour (summer time change)                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul> |

|                                                   |                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"><li>Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR ADD1H LL_RTC_TIME_IncHour</li></ul>                                                            |

### LL\_RTC\_TIME\_GetSubSecond

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_RTC_TIME_GetSubSecond(RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function description                              | Get Sub second value in the synchronous prescaler counter.                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"><li><b>RTCx:</b> RTC Instance</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Return values                                     | <ul style="list-style-type: none"><li><b>Sub:</b> second value (number between 0 and 65535)</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                   |
| Notes                                             | <ul style="list-style-type: none"><li>You can use both SubSeconds value and SecondFraction (PREDIV_S through LL_RTC_GetSynchPrescaler function) terms returned to convert Calendar SubSeconds value in second fraction ratio with time unit following generic formula:<br/>=&gt; Seconds fraction ratio * time_unit= [(SecondFraction-SubSeconds)/(SecondFraction+1)] * time_unit This conversion can be performed only if no shift operation is pending (ie. SHFP=0) when PREDIV_S &gt;= SS.</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>SSR SS LL_RTC_TIME_GetSubSecond</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                         |

### LL\_RTC\_TIME\_Synchronize

|                                     |                                                                                                                                                                                                                                                                                                                                                             |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                       | <code>_STATIC_INLINE void LL_RTC_TIME_Synchronize(RTC_TypeDef * RTCx, uint32_t ShiftSecond, uint32_t Fraction)</code>                                                                                                                                                                                                                                       |
| Function description                | Synchronize to a remote clock with a high degree of precision.                                                                                                                                                                                                                                                                                              |
| Parameters                          | <ul style="list-style-type: none"><li><b>RTCx:</b> RTC Instance</li><li><b>ShiftSecond:</b> This parameter can be one of the following values:<ul style="list-style-type: none"><li>- LL_RTC_SHIFT_SECOND_DELAY</li><li>- LL_RTC_SHIFT_SECOND_ADVANCE</li></ul></li><li><b>Fraction:</b> Number of Seconds Fractions (any value from 0 to 0xFFFF)</li></ul> |
| Return values                       | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                                                                                                                                                                                                                                                 |
| Notes                               | <ul style="list-style-type: none"><li>This operation effectively subtracts from (delays) or advance the clock of a fraction of a second.</li><li>Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li><li>When REFCKON is set, firmware must not write to Shift control register.</li></ul>                          |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"><li>SHIFTR ADD1S LL_RTC_TIME_Synchronize</li></ul>                                                                                                                                                                                                                                                                        |

- 
- reference:
- SHIFTR SUBFS LL\_RTC\_TIME\_Synchronize

### **LL\_RTC\_DATE\_SetYear**

|                                                   |                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_RTC_DATE_SetYear<br/>(RTC_TypeDef * RTCx, uint32_t Year)</code></b>                                                 |
| Function description                              | Set Year in BCD format.                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Year:</b> Value between Min_Data=0x00 and Max_Data=0x99</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Year from binary to BCD format</li> </ul>    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DR YT LL_RTC_DATE_SetYear</li> <li>• DR YU LL_RTC_DATE_SetYear</li> </ul>                                  |

### **LL\_RTC\_DATE\_GetYear**

|                                                   |                                                                                                                                                                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_RTC_DATE_GetYear<br/>(RTC_TypeDef * RTCx)</code></b>                                                                                                                                                                     |
| Function description                              | Get Year in BCD format.                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x99</li> </ul>                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• if shadow mode is disabled (BYPSHAD=0), need to check if RSF flag is set before reading this bit</li> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Year from BCD to Binary format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DR YT LL_RTC_DATE_GetYear</li> <li>• DR YU LL_RTC_DATE_GetYear</li> </ul>                                                                                                                                           |

### **LL\_RTC\_DATE\_SetWeekDay**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_RTC_DATE_SetWeekDay<br/>(RTC_TypeDef * RTCx, uint32_t WeekDay)</code></b>                                                                                                                                                                                                                                                                                                                                            |
| Function description | Set Week day.                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>WeekDay:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_RTC_WEEKDAY_MONDAY</li> <li>- LL_RTC_WEEKDAY_TUESDAY</li> <li>- LL_RTC_WEEKDAY_WEDNESDAY</li> <li>- LL_RTC_WEEKDAY_THURSDAY</li> <li>- LL_RTC_WEEKDAY_FRIDAY</li> <li>- LL_RTC_WEEKDAY_SATURDAY</li> <li>- LL_RTC_WEEKDAY_SUNDAY</li> </ul> </li> </ul> |

|                                                   |                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>DR WDU LL_RTC_DATE_SetWeekDay</li> </ul> |

### LL\_RTC\_DATE\_GetWeekDay

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_DATE_GetWeekDay<br/>(RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Get Week day.                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_WEEKDAY_MONDAY</li> <li>– LL_RTC_WEEKDAY_TUESDAY</li> <li>– LL_RTC_WEEKDAY_WEDNESDAY</li> <li>– LL_RTC_WEEKDAY_THURSDAY</li> <li>– LL_RTC_WEEKDAY_FRIDAY</li> <li>– LL_RTC_WEEKDAY_SATURDAY</li> <li>– LL_RTC_WEEKDAY_SUNDAY</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>if shadow mode is disabled (BYPSHAD=0), need to check if RSF flag is set before reading this bit</li> </ul>                                                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>DR WDU LL_RTC_DATE_GetWeekDay</li> </ul>                                                                                                                                                                                                                                                                                                                                 |

### LL\_RTC\_DATE\_SetMonth

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_DATE_SetMonth<br/>(RTC_TypeDef * RTCx, uint32_t Month)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Set Month in BCD format.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Month:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_MONTH_JANUARY</li> <li>– LL_RTC_MONTH_FEBRUARY</li> <li>– LL_RTC_MONTH_MARCH</li> <li>– LL_RTC_MONTH_APRIIL</li> <li>– LL_RTC_MONTH_MAY</li> <li>– LL_RTC_MONTH_JUNE</li> <li>– LL_RTC_MONTH_JULY</li> <li>– LL_RTC_MONTH_AUGUST</li> <li>– LL_RTC_MONTH_SEPTEMBER</li> <li>– LL_RTC_MONTH_OCTOBER</li> <li>– LL_RTC_MONTH_NOVEMBER</li> <li>– LL_RTC_MONTH_DECEMBER</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>helper macro <code>__LL_RTC_CONVERT_BIN2BCD</code> is available to convert Month from binary to BCD format</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Reference Manual to  | <ul style="list-style-type: none"> <li>DR MT LL_RTC_DATE_SetMonth</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL API cross reference:                     | <ul style="list-style-type: none"> <li>DR MU LL_RTC_DATE_SetMonth</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>LL_RTC_DATE_GetMonth</b>                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function name                               | <code>_STATIC_INLINE uint32_t LL_RTC_DATE_GetMonth (RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                        | Get Month in BCD format.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Return values                               | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_RTC_MONTH_JANUARY</li> <li>- LL_RTC_MONTH_FEBRUARY</li> <li>- LL_RTC_MONTH_MARCH</li> <li>- LL_RTC_MONTH_APRIIL</li> <li>- LL_RTC_MONTH_MAY</li> <li>- LL_RTC_MONTH_JUNE</li> <li>- LL_RTC_MONTH_JULY</li> <li>- LL_RTC_MONTH_AUGUST</li> <li>- LL_RTC_MONTH_SEPTMBER</li> <li>- LL_RTC_MONTH_OCTOBER</li> <li>- LL_RTC_MONTH_NOVEMBER</li> <li>- LL_RTC_MONTH_DECEMBER</li> </ul> </li> </ul> |
| Notes                                       | <ul style="list-style-type: none"> <li>if shadow mode is disabled (BYPSHAD=0), need to check if RSF flag is set before reading this bit</li> <li>helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Month from BCD to Binary format</li> </ul>                                                                                                                                                                                                                                                                                    |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>DR MT LL_RTC_DATE_GetMonth</li> <li>DR MU LL_RTC_DATE_GetMonth</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                             |

## LL\_RTC\_DATE\_SetDay

|                                             |                                                                                                                                                |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE void LL_RTC_DATE_SetDay (RTC_TypeDef * RTCx, uint32_t Day)</code>                                                         |
| Function description                        | Set Day in BCD format.                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Day:</b> Value between Min_Data=0x01 and Max_Data=0x31</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                  |
| Notes                                       | <ul style="list-style-type: none"> <li>helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Day from binary to BCD format</li> </ul>  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>DR DT LL_RTC_DATE_SetDay</li> <li>DR DU LL_RTC_DATE_SetDay</li> </ul>                                   |

**LL\_RTC\_DATE\_GetDay**

|                                                   |                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_RTC_DATE_GetDay<br/>(RTC_TypeDef * RTCx)</code>                                                                                                                                                                            |
| Function description                              | Get Day in BCD format.                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x01 and Max_Data=0x31</li> </ul>                                                                                                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>• if shadow mode is disabled (BYPSHAD=0), need to check if RSF flag is set before reading this bit</li> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Day from BCD to Binary format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DR DT LL_RTC_DATE_GetDay</li> <li>• DR DU LL_RTC_DATE_GetDay</li> </ul>                                                                                                                                            |

**LL\_RTC\_DATE\_Config**

|                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                       | <code>_STATIC_INLINE void LL_RTC_DATE_Config (RTC_TypeDef<br/>* RTCx, uint32_t WeekDay, uint32_t Day, uint32_t Month,<br/>uint32_t Year)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description                | Set date (WeekDay, Day, Month and Year) in BCD format.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                          | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>WeekDay:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_RTC_WEEKDAY_MONDAY</li> <li>- LL_RTC_WEEKDAY_TUESDAY</li> <li>- LL_RTC_WEEKDAY_WEDNESDAY</li> <li>- LL_RTC_WEEKDAY_THURSDAY</li> <li>- LL_RTC_WEEKDAY_FRIDAY</li> <li>- LL_RTC_WEEKDAY_SATURDAY</li> <li>- LL_RTC_WEEKDAY_SUNDAY</li> </ul> </li> <li>• <b>Day:</b> Value between Min_Data=0x01 and Max_Data=0x31</li> <li>• <b>Month:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_RTC_MONTH_JANUARY</li> <li>- LL_RTC_MONTH_FEBRUARY</li> <li>- LL_RTC_MONTH_MARCH</li> <li>- LL_RTC_MONTH_APRIIL</li> <li>- LL_RTC_MONTH_MAY</li> <li>- LL_RTC_MONTH_JUNE</li> <li>- LL_RTC_MONTH_JULY</li> <li>- LL_RTC_MONTH_AUGUST</li> <li>- LL_RTC_MONTH_SEPTEMBER</li> <li>- LL_RTC_MONTH_OCTOBER</li> <li>- LL_RTC_MONTH_NOVEMBER</li> <li>- LL_RTC_MONTH_DECEMBER</li> </ul> </li> <li>• <b>Year:</b> Value between Min_Data=0x00 and Max_Data=0x99</li> </ul> |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• DR WDU LL_RTC_DATE_Config</li> <li>• DR MT LL_RTC_DATE_Config</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

- reference:
- DR MU LL\_RTC\_DATE\_Config
  - DR DT LL\_RTC\_DATE\_Config
  - DR DU LL\_RTC\_DATE\_Config
  - DR YT LL\_RTC\_DATE\_Config
  - DR YU LL\_RTC\_DATE\_Config

### **LL\_RTC\_DATE\_Get**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_DATE_Get (RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                             |
| Function description                              | Get date (WeekDay, Day, Month and Year) in BCD format.                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Combination:</b> of WeekDay, Day, Month and Year (Format: 0xWWDDMMYY).</li> </ul>                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• if shadow mode is disabled (BYPSHAD=0), need to check if RSF flag is set before reading this bit</li> <li>• helper macros <code>__LL_RTC_GET_WEEKDAY</code>, <code>__LL_RTC_GET_YEAR</code>, <code>__LL_RTC_GET_MONTH</code>, and <code>__LL_RTC_GET_DAY</code> are available to get independently each parameter.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DR WDU LL_RTC_DATE_Get</li> <li>• DR MT LL_RTC_DATE_Get</li> <li>• DR MU LL_RTC_DATE_Get</li> <li>• DR DT LL_RTC_DATE_Get</li> <li>• DR DU LL_RTC_DATE_Get</li> <li>• DR YT LL_RTC_DATE_Get</li> <li>• DR YU LL_RTC_DATE_Get</li> </ul>                                                                                       |

### **LL\_RTC\_ALMA\_Enable**

|                                                   |                                                                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_ALMA_Enable (RTC_TypeDef * RTCx)</code>                                                                                |
| Function description                              | Enable Alarm A.                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. <code>LL_RTC_DisableWriteProtection</code> function should be called before.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR ALRAE LL_RTC_ALMA_Enable</li> </ul>                                                                          |

### **LL\_RTC\_ALMA\_Disable**

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_ALMA_Disable (RTC_TypeDef * RTCx)</code>    |
| Function description | Disable Alarm A.                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul> |

|                                                   |                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"><li>None</li></ul>                                                                                    |
| Notes                                             | <ul style="list-style-type: none"><li>Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR ALRAE LL_RTC_ALMA_Disable</li></ul>                                                            |

### LL\_RTC\_ALMA\_SetMask

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_ALMA_SetMask<br/>(RTC_TypeDef * RTCx, uint32_t Mask)</code>                                                                                                                                                                                                                                                                                                                    |
| Function description                              | Specify the Alarm A masks.                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Mask:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>LL_RTC_ALMA_MASK_NONE</li> <li>LL_RTC_ALMA_MASK_DATEWEEKDAY</li> <li>LL_RTC_ALMA_MASK_HOURS</li> <li>LL_RTC_ALMA_MASK_MINUTES</li> <li>LL_RTC_ALMA_MASK_SECONDS</li> <li>LL_RTC_ALMA_MASK_ALL</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"><li>None</li></ul>                                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ALRMAR MSK4 LL_RTC_ALMA_SetMask</li> <li>ALRMAR MSK3 LL_RTC_ALMA_SetMask</li> <li>ALRMAR MSK2 LL_RTC_ALMA_SetMask</li> <li>ALRMAR MSK1 LL_RTC_ALMA_SetMask</li> </ul>                                                                                                                                                                                                    |

### LL\_RTC\_ALMA\_GetMask

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_RTC_ALMA_GetMask<br/>(RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                              |
| Function description                              | Get the Alarm A masks.                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"><li><b>RTCx:</b> RTC Instance</li></ul>                                                                                                                                                                                                                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be can be a combination of the following values: <ul style="list-style-type: none"> <li>LL_RTC_ALMA_MASK_NONE</li> <li>LL_RTC_ALMA_MASK_DATEWEEKDAY</li> <li>LL_RTC_ALMA_MASK_HOURS</li> <li>LL_RTC_ALMA_MASK_MINUTES</li> <li>LL_RTC_ALMA_MASK_SECONDS</li> <li>LL_RTC_ALMA_MASK_ALL</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ALRMAR MSK4 LL_RTC_ALMA_GetMask</li> <li>ALRMAR MSK3 LL_RTC_ALMA_GetMask</li> <li>ALRMAR MSK2 LL_RTC_ALMA_GetMask</li> <li>ALRMAR MSK1 LL_RTC_ALMA_GetMask</li> </ul>                                                                                                                                                                   |

**LL\_RTC\_ALMA\_EnableWeekday**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_RTC_ALMA_EnableWeekday<br/>(RTC_TypeDef * RTCx)</b>              |
| Function description                              | Enable AlarmA Week day selection (DU[3:0] represents the week day).                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMAR WDSEL LL_RTC_ALMA_EnableWeekday</li> </ul> |

**LL\_RTC\_ALMA\_DisableWeekday**

|                                                   |                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_RTC_ALMA_DisableWeekday<br/>(RTC_TypeDef * RTCx)</b>              |
| Function description                              | Disable AlarmA Week day selection (DU[3:0] represents the date )                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMAR WDSEL LL_RTC_ALMA_DisableWeekday</li> </ul> |

**LL\_RTC\_ALMA\_SetDay**

|                                                   |                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_RTC_ALMA_SetDay (RTC_TypeDef<br/>* RTCx, uint32_t Day)</b>                                                               |
| Function description                              | Set ALARM A Day in BCD format.                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Day:</b> Value between Min_Data=0x01 and Max_Data=0x31</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Day from binary to BCD format</li> </ul>    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMAR DT LL_RTC_ALMA_SetDay</li> <li>• ALRMAR DU LL_RTC_ALMA_SetDay</li> </ul>                           |

**LL\_RTC\_ALMA\_GetDay**

|                      |                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE uint32_t LL_RTC_ALMA_GetDay<br/>(RTC_TypeDef * RTCx)</b>                                |
| Function description | Get ALARM A Day in BCD format.                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x01 and Max_Data=0x31</li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available</li> </ul>    |

to convert Day from BCD to Binary format

Reference Manual to  
LL API cross  
reference:

- ALRMAR DT LL\_RTC\_ALMA\_GetDay
- ALRMAR DU LL\_RTC\_ALMA\_GetDay

### **LL\_RTC\_ALMA\_SetWeekDay**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_RTC_ALMA_SetWeekDay(<br/>(RTC_TypeDef * RTCx, uint32_t WeekDay)</code></b>                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Set ALARM A Weekday.                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>WeekDay:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_RTC_WEEKDAY_MONDAY</li> <li>- LL_RTC_WEEKDAY_TUESDAY</li> <li>- LL_RTC_WEEKDAY_WEDNESDAY</li> <li>- LL_RTC_WEEKDAY_THURSDAY</li> <li>- LL_RTC_WEEKDAY_FRIDAY</li> <li>- LL_RTC_WEEKDAY_SATURDAY</li> <li>- LL_RTC_WEEKDAY_SUNDAY</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMAR DU LL_RTC_ALMA_SetWeekDay</li> </ul>                                                                                                                                                                                                                                                                                                                                                                 |

### **LL\_RTC\_ALMA\_GetWeekDay**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RTC_ALMA_GetWeekDay(<br/>(RTC_TypeDef * RTCx)</code></b>                                                                                                                                                                                                                                                                                                           |
| Function description                              | Get ALARM A Weekday.                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_RTC_WEEKDAY_MONDAY</li> <li>- LL_RTC_WEEKDAY_TUESDAY</li> <li>- LL_RTC_WEEKDAY_WEDNESDAY</li> <li>- LL_RTC_WEEKDAY_THURSDAY</li> <li>- LL_RTC_WEEKDAY_FRIDAY</li> <li>- LL_RTC_WEEKDAY_SATURDAY</li> <li>- LL_RTC_WEEKDAY_SUNDAY</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMAR DU LL_RTC_ALMA_GetWeekDay</li> </ul>                                                                                                                                                                                                                                                                                                                    |

### **LL\_RTC\_ALMA\_SetTimeFormat**

|                      |                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_RTC_ALMA_SetTimeFormat(<br/>(RTC_TypeDef * RTCx, uint32_t TimeFormat)</code></b> |
| Function description | Set Alarm A time format (AM/24-hour or PM notation)                                                               |

|                                                   |                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>TimeFormat:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_ALMA_TIME_FORMAT_AM</li> <li>– LL_RTC_ALMA_TIME_FORMAT_PM</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ALRMAR PM LL_RTC_ALMA_SetTimeFormat</li> </ul>                                                                                                                                                                                                      |

### LL\_RTC\_ALMA\_GetTimeFormat

|                                                   |                                                                                                                                                                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_RTC_ALMA_GetTimeFormat(<br/>(RTC_TypeDef * RTCx)</code>                                                                                                                                                     |
| Function description                              | Get Alarm A time format (AM or PM notation)                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_ALMA_TIME_FORMAT_AM</li> <li>– LL_RTC_ALMA_TIME_FORMAT_PM</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ALRMAR PM LL_RTC_ALMA_GetTimeFormat</li> </ul>                                                                                                                                                        |

### LL\_RTC\_ALMA\_SetHour

|                                                   |                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_ALMA_SetHour(<br/>(RTC_TypeDef * RTCx, uint32_t Hours)</code>                                                                                              |
| Function description                              | Set ALARM A Hours in BCD format.                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Hours:</b> Value between Min_Data=0x01 and Max_Data=0x12 or between Min_Data=0x00 and Max_Data=0x23</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Hours from binary to BCD format</li> </ul>                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ALRMAR HT LL_RTC_ALMA_SetHour</li> <li>ALRMAR HU LL_RTC_ALMA_SetHour</li> </ul>                                                                      |

### LL\_RTC\_ALMA\_GetHour

|                      |                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_RTC_ALMA_GetHour(<br/>(RTC_TypeDef * RTCx)</code>                                                                 |
| Function description | Get ALARM A Hours in BCD format.                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                        |
| Return values        | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x01 and Max_Data=0x12 or between Min_Data=0x00 and Max_Data=0x23</li> </ul> |

|                                             |                                                                                                                                                 |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Hours from BCD to Binary format</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>ALRMAR HT LL_RTC_ALMA_GetHour</li> <li>ALRMAR HU LL_RTC_ALMA_GetHour</li> </ul>                          |

### LL\_RTC\_ALMA\_SetMinute

|                                             |                                                                                                                                                    |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_RTC_ALMA_SetMinute(RTC_TypeDef * RTCx, uint32_t Minutes)</code></b>                                                |
| Function description                        | Set ALARM A Minutes in BCD format.                                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Minutes:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                      |
| Notes                                       | <ul style="list-style-type: none"> <li>helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Minutes from binary to BCD format</li> </ul>  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>ALRMAR MNT LL_RTC_ALMA_SetMinute</li> <li>ALRMAR MNU LL_RTC_ALMA_SetMinute</li> </ul>                       |

### LL\_RTC\_ALMA\_GetMinute

|                                             |                                                                                                                                                   |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_RTC_ALMA_GetMinute(RTC_TypeDef * RTCx)</code></b>                                                             |
| Function description                        | Get ALARM A Minutes in BCD format.                                                                                                                |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                       |
| Return values                               | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x00 and Max_Data=0x59</li> </ul>                                           |
| Notes                                       | <ul style="list-style-type: none"> <li>helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Minutes from BCD to Binary format</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>ALRMAR MNT LL_RTC_ALMA_GetMinute</li> <li>ALRMAR MNU LL_RTC_ALMA_GetMinute</li> </ul>                      |

### LL\_RTC\_ALMA\_SetSecond

|                                             |                                                                                                                                                    |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_RTC_ALMA_SetSecond(RTC_TypeDef * RTCx, uint32_t Seconds)</code></b>                                                |
| Function description                        | Set ALARM A Seconds in BCD format.                                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Seconds:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                      |
| Notes                                       | <ul style="list-style-type: none"> <li>helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Seconds from binary to BCD format</li> </ul>  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>ALRMAR ST LL_RTC_ALMA_SetSecond</li> </ul>                                                                  |

- ALRMAR SU LL\_RTC\_ALMA\_SetSecond
- LL API cross reference:

### LL\_RTC\_ALMA\_GetSecond

|                                             |                                                                                                                                                     |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_RTC_ALMA_GetSecond<br/>(RTC_TypeDef * RTCx)</code>                                                                |
| Function description                        | Get ALARM A Seconds in BCD format.                                                                                                                  |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                       |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x59</li> </ul>                                           |
| Notes                                       | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Seconds from BCD to Binary format</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• ALRMAR ST LL_RTC_ALMA_GetSecond</li> <li>• ALRMAR SU LL_RTC_ALMA_GetSecond</li> </ul>                      |

### LL\_RTC\_ALMA\_ConfigTime

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_RTC_ALMA_ConfigTime<br/>(RTC_TypeDef * RTCx, uint32_t Format12_24, uint32_t Hours,<br/>uint32_t Minutes, uint32_t Seconds)</code>                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                        | Set Alarm A Time (hour, minute and second) in BCD format.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Format12_24:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_RTC_ALMA_TIME_FORMAT_AM</li> <li>- LL_RTC_ALMA_TIME_FORMAT_PM</li> </ul> </li> <li>• <b>Hours:</b> Value between Min_Data=0x01 and Max_Data=0x12 or between Min_Data=0x00 and Max_Data=0x23</li> <li>• <b>Minutes:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> <li>• <b>Seconds:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• ALRMAR PM LL_RTC_ALMA_ConfigTime</li> <li>• ALRMAR HT LL_RTC_ALMA_ConfigTime</li> <li>• ALRMAR HU LL_RTC_ALMA_ConfigTime</li> <li>• ALRMAR MNT LL_RTC_ALMA_ConfigTime</li> <li>• ALRMAR MNU LL_RTC_ALMA_ConfigTime</li> <li>• ALRMAR ST LL_RTC_ALMA_ConfigTime</li> <li>• ALRMAR SU LL_RTC_ALMA_ConfigTime</li> </ul>                                                                                                                                                                                            |

### LL\_RTC\_ALMA\_GetTime

|                      |                                                                                    |
|----------------------|------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_RTC_ALMA_GetTime<br/>(RTC_TypeDef * RTCx)</code> |
| Function description | Get Alarm B Time (hour, minute and second) in BCD format.                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>      |

|                                                   |                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>Combination:</b> of hours, minutes and seconds.</li> </ul>                                                                                                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>helper macros <code>_LL_RTC_GET_HOUR</code>, <code>_LL_RTC_GET_MINUTE</code> and <code>_LL_RTC_GET_SECOND</code> are available to get independently each parameter.</li> </ul>                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ALRMAR HT LL_RTC_ALMA_GetTime</li> <li>ALRMAR HU LL_RTC_ALMA_GetTime</li> <li>ALRMAR MNT LL_RTC_ALMA_GetTime</li> <li>ALRMAR MNU LL_RTC_ALMA_GetTime</li> <li>ALRMAR ST LL_RTC_ALMA_GetTime</li> <li>ALRMAR SU LL_RTC_ALMA_GetTime</li> </ul> |

### LL\_RTC\_ALMA\_SetSubSecondMask

|                                                   |                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_ALMA_SetSubSecondMask (RTC_TypeDef * RTCx, uint32_t Mask)</code>                                                      |
| Function description                              | Set Alarm A Mask the most-significant bits starting at this bit.                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Mask:</b> Value between Min_Data=0x00 and Max_Data=0xF</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>This register can be written only when ALRAE is reset in RTC_CR register, or in initialization mode.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ALRMASSR MASKSS LL_RTC_ALMA_SetSubSecondMask</li> </ul>                                                         |

### LL\_RTC\_ALMA\_GetSubSecondMask

|                                                   |                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_RTC_ALMA_GetSubSecondMask (RTC_TypeDef * RTCx)</code>                 |
| Function description                              | Get Alarm A Mask the most-significant bits starting at this bit.                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                            |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x00 and Max_Data=0xF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ALRMASSR MASKSS LL_RTC_ALMA_GetSubSecondMask</li> </ul>         |

### LL\_RTC\_ALMA\_SetSubSecond

|                      |                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_RTC_ALMA_SetSubSecond (RTC_TypeDef * RTCx, uint32_t Subsecond)</code>                                                     |
| Function description | Set Alarm A Sub seconds value.                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Subsecond:</b> Value between Min_Data=0x00 and Max_Data=0x7FFF</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                          |
| Reference Manual to  | <ul style="list-style-type: none"> <li>ALRMASSR SS LL_RTC_ALMA_SetSubSecond</li> </ul>                                                                 |

LL API cross  
reference:

### **LL\_RTC\_ALMA\_GetSubSecond**

Function name      **\_STATIC\_INLINE uint32\_t LL\_RTC\_ALMA\_GetSubSecond (RTC\_TypeDef \* RTCx)**

Function description      Get Alarm A Sub seconds value.

Parameters      • **RTCx:** RTC Instance

Return values      • **Value:** between Min\_Data=0x00 and Max\_Data=0x7FFF

Reference Manual to  
LL API cross  
reference:

### **LL\_RTC\_ALMB\_Enable**

Function name      **\_STATIC\_INLINE void LL\_RTC\_ALMB\_Enable (RTC\_TypeDef \* RTCx)**

Function description      Enable Alarm B.

Parameters      • **RTCx:** RTC Instance

Return values      • **None**

Notes      • Bit is write-protected. LL\_RTC\_DisableWriteProtection function should be called before.

Reference Manual to  
LL API cross  
reference:

### **LL\_RTC\_ALMB\_Disable**

Function name      **\_STATIC\_INLINE void LL\_RTC\_ALMB\_Disable (RTC\_TypeDef \* RTCx)**

Function description      Disable Alarm B.

Parameters      • **RTCx:** RTC Instance

Return values      • **None**

Notes      • Bit is write-protected. LL\_RTC\_DisableWriteProtection function should be called before.

Reference Manual to  
LL API cross  
reference:

### **LL\_RTC\_ALMB\_SetMask**

Function name      **\_STATIC\_INLINE void LL\_RTC\_ALMB\_SetMask (RTC\_TypeDef \* RTCx, uint32\_t Mask)**

Function description      Specify the Alarm B masks.

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Mask:</b> This parameter can be a combination of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_ALMB_MASK_NONE</li> <li>- LL_RTC_ALMB_MASK_DATEWEEKDAY</li> <li>- LL_RTC_ALMB_MASK_HOURS</li> <li>- LL_RTC_ALMB_MASK_MINUTES</li> <li>- LL_RTC_ALMB_MASK_SECONDS</li> <li>- LL_RTC_ALMB_MASK_ALL</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR MSK4 LL_RTC_ALMB_SetMask</li> <li>• ALRMBR MSK3 LL_RTC_ALMB_SetMask</li> <li>• ALRMBR MSK2 LL_RTC_ALMB_SetMask</li> <li>• ALRMBR MSK1 LL_RTC_ALMB_SetMask</li> </ul>                                                                                                                                                                                                                  |

### LL\_RTC\_ALMB\_GetMask

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_RTC_ALMB_GetMask(<br/>(RTC_TypeDef * RTCx))</code>                                                                                                                                                                                                                                                                                                                  |
| Function description                              | Get the Alarm B masks.                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be can be a combination of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_ALMB_MASK_NONE</li> <li>- LL_RTC_ALMB_MASK_DATEWEEKDAY</li> <li>- LL_RTC_ALMB_MASK_HOURS</li> <li>- LL_RTC_ALMB_MASK_MINUTES</li> <li>- LL_RTC_ALMB_MASK_SECONDS</li> <li>- LL_RTC_ALMB_MASK_ALL</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR MSK4 LL_RTC_ALMB_GetMask</li> <li>• ALRMBR MSK3 LL_RTC_ALMB_GetMask</li> <li>• ALRMBR MSK2 LL_RTC_ALMB_GetMask</li> <li>• ALRMBR MSK1 LL_RTC_ALMB_GetMask</li> </ul>                                                                                                                                                                                 |

### LL\_RTC\_ALMB\_EnableWeekday

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_ALMB_EnableWeekday(<br/>(RTC_TypeDef * RTCx))</code>      |
| Function description                              | Enable AlarmB Week day selection (DU[3:0] represents the week day.                         |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR WDSEL LL_RTC_ALMB_EnableWeekday</li> </ul> |

**LL\_RTC\_ALMB\_DisableWeekday**

|                                                   |                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_ALMB_DisableWeekday<br/>(RTC_TypeDef * RTCx)</code>       |
| Function description                              | Disable AlarmB Week day selection (DU[3:0] represents the date )                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR WDSEL LL_RTC_ALMB_DisableWeekday</li> </ul> |

**LL\_RTC\_ALMB\_SetDay**

|                                                   |                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_ALMB_SetDay (RTC_TypeDef<br/>* RTCx, uint32_t Day)</code>                                                        |
| Function description                              | Set ALARM B Day in BCD format.                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Day:</b> Value between Min_Data=0x01 and Max_Data=0x31</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Day from binary to BCD format</li> </ul>    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR DT LL_RTC_ALMB_SetDay</li> <li>• ALRMBR DU LL_RTC_ALMB_SetDay</li> </ul>                           |

**LL\_RTC\_ALMB\_GetDay**

|                                                   |                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_ALMB_GetDay<br/>(RTC_TypeDef * RTCx)</code>                                                               |
| Function description                              | Get ALARM B Day in BCD format.                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x01 and Max_Data=0x31</li> </ul>                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Day from BCD to Binary format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR DT LL_RTC_ALMB_GetDay</li> <li>• ALRMBR DU LL_RTC_ALMB_GetDay</li> </ul>                        |

**LL\_RTC\_ALMB\_SetWeekDay**

|                      |                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_ALMB_SetWeekDay<br/>(RTC_TypeDef * RTCx, uint32_t WeekDay)</code>                                                         |
| Function description | Set ALARM B Weekday.                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>WeekDay:</b> This parameter can be one of the following values:</li> </ul> |

- LL\_RTC\_WEEKDAY\_MONDAY
- LL\_RTC\_WEEKDAY\_TUESDAY
- LL\_RTC\_WEEKDAY\_WEDNESDAY
- LL\_RTC\_WEEKDAY\_THURSDAY
- LL\_RTC\_WEEKDAY\_FRIDAY
- LL\_RTC\_WEEKDAY\_SATURDAY
- LL\_RTC\_WEEKDAY\_SUNDAY

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- ALRMBR DU LL\_RTC\_ALMB\_SetWeekDay

### **LL\_RTC\_ALMB\_GetWeekDay**

Function name      **STATIC\_INLINE uint32\_t LL\_RTC\_ALMB\_GetWeekDay(RTC\_TypeDef \* RTCx)**

Function description      Get ALARM B Weekday.

Parameters      • **RTCx:** RTC Instance

Return values      • **Returned:** value can be one of the following values:

- LL\_RTC\_WEEKDAY\_MONDAY
- LL\_RTC\_WEEKDAY\_TUESDAY
- LL\_RTC\_WEEKDAY\_WEDNESDAY
- LL\_RTC\_WEEKDAY\_THURSDAY
- LL\_RTC\_WEEKDAY\_FRIDAY
- LL\_RTC\_WEEKDAY\_SATURDAY
- LL\_RTC\_WEEKDAY\_SUNDAY

Reference Manual to  
LL API cross  
reference:

- ALRMBR DU LL\_RTC\_ALMB\_GetWeekDay

### **LL\_RTC\_ALMB\_SetTimeFormat**

Function name      **STATIC\_INLINE void LL\_RTC\_ALMB\_SetTimeFormat(RTC\_TypeDef \* RTCx, uint32\_t TimeFormat)**

Function description      Set ALARM B time format (AM/24-hour or PM notation)

Parameters

- **RTCx:** RTC Instance
- **TimeFormat:** This parameter can be one of the following values:
  - LL\_RTC\_ALMB\_TIME\_FORMAT\_AM
  - LL\_RTC\_ALMB\_TIME\_FORMAT\_PM

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- ALRMBR PM LL\_RTC\_ALMB\_SetTimeFormat

**LL\_RTC\_ALMB\_GetTimeFormat**

|                                                   |                                                                                                                                                                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_RTC_ALMB_GetTimeFormat<br/>(RTC_TypeDef * RTCx)</code>                                                                                                                                                        |
| Function description                              | Get ALARM B time format (AM or PM notation)                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_ALMB_TIME_FORMAT_AM</li> <li>- LL_RTC_ALMB_TIME_FORMAT_PM</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR PM LL_RTC_ALMB_GetTimeFormat</li> </ul>                                                                                                                                                        |

**LL\_RTC\_ALMB\_SetHour**

|                                                   |                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_ALMB_SetHour<br/>(RTC_TypeDef * RTCx, uint32_t Hours)</code>                                                                                                   |
| Function description                              | Set ALARM B Hours in BCD format.                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Hours:</b> Value between Min_Data=0x01 and Max_Data=0x12 or between Min_Data=0x00 and Max_Data=0x23</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Hours from binary to BCD format</li> </ul>                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR HT LL_RTC_ALMB_SetHour</li> <li>• ALRMBR HU LL_RTC_ALMB_SetHour</li> </ul>                                                                      |

**LL\_RTC\_ALMB\_GetHour**

|                                                   |                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_RTC_ALMB_GetHour<br/>(RTC_TypeDef * RTCx)</code>                                                                    |
| Function description                              | Get ALARM B Hours in BCD format.                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x01 and Max_Data=0x12 or between Min_Data=0x00 and Max_Data=0x23</li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Hours from BCD to Binary format</li> </ul>    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR HT LL_RTC_ALMB_GetHour</li> <li>• ALRMBR HU LL_RTC_ALMB_GetHour</li> </ul>                           |

**LL\_RTC\_ALMB\_SetMinute**

|                                                   |                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_ALMB_SetMinute<br/>(RTC_TypeDef * RTCx, uint32_t Minutes)</code>                                                  |
| Function description                              | Set ALARM B Minutes in BCD format.                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Minutes:</b> between Min_Data=0x00 and Max_Data=0x59</li> </ul>    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Minutes from binary to BCD format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR MNT LL_RTC_ALMB_SetMinute</li> <li>• ALRMBR MNU LL_RTC_ALMB_SetMinute</li> </ul>                    |

**LL\_RTC\_ALMB\_GetMinute**

|                                                   |                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_ALMB_GetMinute<br/>(RTC_TypeDef * RTCx)</code>                                                                |
| Function description                              | Get ALARM B Minutes in BCD format.                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x59</li> </ul>                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Minutes from BCD to Binary format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR MNT LL_RTC_ALMB_GetMinute</li> <li>• ALRMBR MNU LL_RTC_ALMB_GetMinute</li> </ul>                    |

**LL\_RTC\_ALMB\_SetSecond**

|                                                   |                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_ALMB_SetSecond<br/>(RTC_TypeDef * RTCx, uint32_t Seconds)</code>                                                     |
| Function description                              | Set ALARM B Seconds in BCD format.                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Seconds:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BIN2BCD is available to convert Seconds from binary to BCD format</li> </ul>    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR ST LL_RTC_ALMB_SetSecond</li> <li>• ALRMBR SU LL_RTC_ALMB_SetSecond</li> </ul>                         |

**LL\_RTC\_ALMB\_GetSecond**

|                                                   |                                                                                                                                                                  |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_ALMB_GetSecond<br/>(RTC_TypeDef * RTCx)</code>                                                                             |
| Function description                              | Get ALARM B Seconds in BCD format.                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x59</li> </ul>                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro <code>__LL_RTC_CONVERT_BCD2BIN</code> is available to convert Seconds from BCD to Binary format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR ST LL_RTC_ALMB_GetSecond</li> <li>• ALRMBR SU LL_RTC_ALMB_GetSecond</li> </ul>                                   |

**LL\_RTC\_ALMB\_ConfigTime**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_ALMB_ConfigTime<br/>(RTC_TypeDef * RTCx, uint32_t Format12_24, uint32_t Hours,<br/>uint32_t Minutes, uint32_t Seconds)</code>                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description                              | Set Alarm B Time (hour, minute and second) in BCD format.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Format12_24:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_RTC_ALMB_TIME_FORMAT_AM</code></li> <li>- <code>LL_RTC_ALMB_TIME_FORMAT_PM</code></li> </ul> </li> <li>• <b>Hours:</b> Value between Min_Data=0x01 and Max_Data=0x12 or between Min_Data=0x00 and Max_Data=0x23</li> <li>• <b>Minutes:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> <li>• <b>Seconds:</b> Value between Min_Data=0x00 and Max_Data=0x59</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBR PM LL_RTC_ALMB_ConfigTime</li> <li>• ALRMBR HT LL_RTC_ALMB_ConfigTime</li> <li>• ALRMBR HU LL_RTC_ALMB_ConfigTime</li> <li>• ALRMBR MNT LL_RTC_ALMB_ConfigTime</li> <li>• ALRMBR MNU LL_RTC_ALMB_ConfigTime</li> <li>• ALRMBR ST LL_RTC_ALMB_ConfigTime</li> <li>• ALRMBR SU LL_RTC_ALMB_ConfigTime</li> </ul>                                                                                                                                                                                                                      |

**LL\_RTC\_ALMB\_GetTime**

|                      |                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_RTC_ALMB_GetTime<br/>(RTC_TypeDef * RTCx)</code>                     |
| Function description | Get Alarm B Time (hour, minute and second) in BCD format.                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Combination:</b> of hours, minutes and seconds.</li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• helper macros <code>__LL_RTC_GET_HOUR</code>,</li> </ul>      |

`__LL_RTC_GET_MINUTE` and `__LL_RTC_GET_SECOND` are available to get independently each parameter.

Reference Manual to  
LL API cross  
reference:

- ALRMBR HT LL\_RTC\_ALMB\_GetTime
- ALRMBR HU LL\_RTC\_ALMB\_GetTime
- ALRMBR MNT LL\_RTC\_ALMB\_GetTime
- ALRMBR MNU LL\_RTC\_ALMB\_GetTime
- ALRMBR ST LL\_RTC\_ALMB\_GetTime
- ALRMBR SU LL\_RTC\_ALMB\_GetTime

### LL\_RTC\_ALMB\_SetSubSecondMask

|                                                   |                                                                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_ALMB_SetSubSecondMask (RTC_TypeDef * RTCx, uint32_t Mask)</code>                                                       |
| Function description                              | Set Alarm B Mask the most-significant bits starting at this bit.                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Mask:</b> Value between Min_Data=0x00 and Max_Data=0xF</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• This register can be written only when ALRBE is reset in RTC_CR register, or in initialization mode.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBSSR MASKSS LL_RTC_ALMB_SetSubSecondMask</li> </ul>                                                         |

### LL\_RTC\_ALMB\_GetSubSecondMask

|                      |                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_RTC_ALMB_GetSubSecondMask (RTC_TypeDef * RTCx)</code>                  |
| Function description | Get Alarm B Mask the most-significant bits starting at this bit.                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                            |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xF</li> </ul> |

Reference Manual to  
LL API cross  
reference:

### LL\_RTC\_ALMB\_SetSubSecond

|                                                   |                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_ALMB_SetSubSecond (RTC_TypeDef * RTCx, uint32_t Subsecond)</code>                                                        |
| Function description                              | Set Alarm B Sub seconds value.                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Subsecond:</b> Value between Min_Data=0x00 and Max_Data=0x7FFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBSSR SS LL_RTC_ALMB_SetSubSecond</li> </ul>                                                                   |

**LL\_RTC\_ALMB\_GetSubSecond**

|                                                   |                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_RTC_ALMB_GetSubSecond (RTC_TypeDef * RTCx)</code>                          |
| Function description                              | Get Alarm B Sub seconds value.                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x7FFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ALRMBSSR SS LL_RTC_ALMB_GetSubSecond</li> </ul>                    |

**LL\_RTC\_TS\_Enable**

|                                                   |                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_TS_Enable (RTC_TypeDef * RTCx)</code>                                                                      |
| Function description                              | Enable Timestamp.                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR TSE LL_RTC_TS_Enable</li> </ul>                                                                 |

**LL\_RTC\_TS\_Disable**

|                                                   |                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_TS_Disable (RTC_TypeDef * RTCx)</code>                                                                     |
| Function description                              | Disable Timestamp.                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR TSE LL_RTC_TS_Disable</li> </ul>                                                                |

**LL\_RTC\_TS\_SetActiveEdge**

|                      |                                                                                                                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_RTC_TS_SetActiveEdge (RTC_TypeDef * RTCx, uint32_t Edge)</code>                                                                                                                                                                                        |
| Function description | Set Time-stamp event active edge.                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Edge:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_RTC_TIMESTAMP_EDGE_RISING</li> <li>- LL_RTC_TIMESTAMP_EDGE_FALLING</li> </ul> </li> </ul> |

---

|                                                   |                                                                                                                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> <li>TSE must be reset when TSEDGE is changed to avoid unwanted TSF setting</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR TSEDGE LL_RTC_TS_SetActiveEdge</li> </ul>                                                                                                                                       |

### LL\_RTC\_TS\_GetActiveEdge

|                                                   |                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_RTC_TS_GetActiveEdge(RTC_TypeDef * RTCx)</code></b>                                                                                                                                                           |
| Function description                              | Get Time-stamp event active edge.                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_TIMESTAMP_EDGE_RISING</li> <li>- LL_RTC_TIMESTAMP_EDGE_FALLING</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul>                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR TSEDGE LL_RTC_TS_GetActiveEdge</li> </ul>                                                                                                                                                               |

### LL\_RTC\_TS\_GetTimeFormat

|                                                   |                                                                                                                                                                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_RTC_TS_GetTimeFormat(RTC_TypeDef * RTCx)</code></b>                                                                                                                                                  |
| Function description                              | Get Timestamp AM/PM notation (AM or 24-hour format)                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                              |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_TS_TIME_FORMAT_AM</li> <li>- LL_RTC_TS_TIME_FORMAT_PM</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TSTR PM LL_RTC_TS_GetTimeFormat</li> </ul>                                                                                                                                                        |

### LL\_RTC\_TS\_GetHour

|                      |                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_RTC_TS_GetHour(RTC_TypeDef * RTCx)</code></b>                                                                  |
| Function description | Get Timestamp Hours in BCD format.                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                        |
| Return values        | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x01 and Max_Data=0x12 or between Min_Data=0x00 and Max_Data=0x23</li> </ul> |

Notes

- helper macro `_LL_RTC_CONVERT_BCD2BIN` is available

to convert Hours from BCD to Binary format

Reference Manual to  
LL API cross  
reference:

- TSTR HT LL\_RTC\_TS\_GetHour
- TSTR HU LL\_RTC\_TS\_GetHour

### **LL\_RTC\_TS\_GetMinute**

|                                                   |                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RTC_TS_GetMinute(<br/>(RTC_TypeDef * RTCx)</code></b>                                                          |
| Function description                              | Get Timestamp Minutes in BCD format.                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x59</li> </ul>                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Minutes from BCD to Binary format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TSTR MNT LL_RTC_TS_GetMinute</li> <li>• TSTR MNU LL_RTC_TS_GetMinute</li> </ul>                            |

### **LL\_RTC\_TS\_GetSecond**

|                                                   |                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RTC_TS_GetSecond(<br/>(RTC_TypeDef * RTCx)</code></b>                                                          |
| Function description                              | Get Timestamp Seconds in BCD format.                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x59</li> </ul>                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro __LL_RTC_CONVERT_BCD2BIN is available to convert Seconds from BCD to Binary format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TSTR ST LL_RTC_TS_GetSecond</li> <li>• TSTR SU LL_RTC_TS_GetSecond</li> </ul>                              |

### **LL\_RTC\_TS\_GetTime**

|                                                   |                                                                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RTC_TS_GetTime(<br/>(RTC_TypeDef * RTCx)</code></b>                                                                                                                                            |
| Function description                              | Get Timestamp time (hour, minute and second) in BCD format.                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Combination:</b> of hours, minutes and seconds.</li> </ul>                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macros __LL_RTC_GET_HOUR,<br/>__LL_RTC_GET_MINUTE and __LL_RTC_GET_SECOND are available to get independently each parameter.</li> </ul>                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TSTR HT LL_RTC_TS_GetTime</li> <li>• TSTR HU LL_RTC_TS_GetTime</li> <li>• TSTR MNT LL_RTC_TS_GetTime</li> <li>• TSTR MNU LL_RTC_TS_GetTime</li> <li>• TSTR ST LL_RTC_TS_GetTime</li> </ul> |

- TSTR SU LL\_RTC\_TS\_GetTime

### **LL\_RTC\_TS\_GetWeekDay**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_TS_GetWeekDay(<br/>    RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Get Timestamp Week day.                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_WEEKDAY_MONDAY</li> <li>- LL_RTC_WEEKDAY_TUESDAY</li> <li>- LL_RTC_WEEKDAY_WEDNESDAY</li> <li>- LL_RTC_WEEKDAY_THURSDAY</li> <li>- LL_RTC_WEEKDAY_FRIDAY</li> <li>- LL_RTC_WEEKDAY_SATURDAY</li> <li>- LL_RTC_WEEKDAY_SUNDAY</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TSDR WDU LL_RTC_TS_GetWeekDay</li> </ul>                                                                                                                                                                                                                                                                                                                                 |

### **LL\_RTC\_TS\_GetMonth**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_TS_GetMonth(<br/>    RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description                              | Get Timestamp Month in BCD format.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_MONTH_JANUARY</li> <li>- LL_RTC_MONTH_FEBRUARY</li> <li>- LL_RTC_MONTH_MARCH</li> <li>- LL_RTC_MONTH_APRI</li> <li>- LL_RTC_MONTH_MAY</li> <li>- LL_RTC_MONTH_JUNE</li> <li>- LL_RTC_MONTH_JULY</li> <li>- LL_RTC_MONTH_AUGUST</li> <li>- LL_RTC_MONTH_SEPTMBER</li> <li>- LL_RTC_MONTH_OCTOBER</li> <li>- LL_RTC_MONTH_NOVEMBER</li> <li>- LL_RTC_MONTH_DECEMBER</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro <code>__LL_RTC_CONVERT_BCD2BIN</code> is available to convert Month from BCD to Binary format</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TSDR MT LL_RTC_TS_GetMonth</li> <li>• TSDR MU LL_RTC_TS_GetMonth</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                   |

**LL\_RTC\_TS\_GetDay**

|                                                   |                                                                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_TS_GetDay<br/>(RTC_TypeDef * RTCx)</code>                                                                              |
| Function description                              | Get Timestamp Day in BCD format.                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x01 and Max_Data=0x31</li> </ul>                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macro <code>__LL_RTC_CONVERT_BCD2BIN</code> is available to convert Day from BCD to Binary format</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TSDR DT LL_RTC_TS_GetDay</li> <li>• TSDR DU LL_RTC_TS_GetDay</li> </ul>                                             |

**LL\_RTC\_TS\_GetDate**

|                                                   |                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_TS_GetDate<br/>(RTC_TypeDef * RTCx)</code>                                                                                                                                                   |
| Function description                              | Get Timestamp date (WeekDay, Day and Month) in BCD format.                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Combination:</b> of Weekday, Day and Month</li> </ul>                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• helper macros <code>__LL_RTC_GET_WEEKDAY</code>, <code>__LL_RTC_GET_MONTH</code>, and <code>__LL_RTC_GET_DAY</code> are available to get independently each parameter.</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TSDR WDU LL_RTC_TS_GetDate</li> <li>• TSDR MT LL_RTC_TS_GetDate</li> <li>• TSDR MU LL_RTC_TS_GetDate</li> <li>• TSDR DT LL_RTC_TS_GetDate</li> <li>• TSDR DU LL_RTC_TS_GetDate</li> </ul> |

**LL\_RTC\_TS\_GetSubSecond**

|                                                   |                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_TS_GetSubSecond<br/>(RTC_TypeDef * RTCx)</code>                       |
| Function description                              | Get time-stamp sub second value.                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xFFFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TSSSR SS LL_RTC_TS_GetSubSecond</li> </ul>                         |

**LL\_RTC\_TS\_EnableOnTamper**

|                      |                                                                                     |
|----------------------|-------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_TS_EnableOnTamper<br/>(RTC_TypeDef * RTCx)</code> |
| Function description | Activate timestamp on tamper detection event.                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>       |

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TAFCR TAMPTS LL_RTC_TS_EnableOnTamper</li> </ul> |

### LL\_RTC\_TS\_DisableOnTamper

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_TS_DisableOnTamper(<br/>    RTC_TypeDef * RTCx)</code> |
| Function description                              | Disable timestamp on tamper detection event.                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TAFCR TAMPTS LL_RTC_TS_DisableOnTamper</li> </ul> |

### LL\_RTC\_TAMPER\_Enable

|                                                   |                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_TAMPER_Enable(<br/>    RTC_TypeDef * RTCx, uint32_t Tamper)</code>                                                                                                                                                                                                                                              |
| Function description                              | Enable RTC_TAMPx input detection.                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Tamper:</b> This parameter can be a combination of the following values: (*) value not defined in all devices.             <ul style="list-style-type: none"> <li>– LL_RTC_TAMPER_1</li> <li>– LL_RTC_TAMPER_2</li> <li>– LL_RTC_TAMPER_3 (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TAFCR TAMP1E LL_RTC_TAMPER_Enable</li> <li>TAFCR TAMP2E LL_RTC_TAMPER_Enable</li> <li>TAFCR TAMP3E LL_RTC_TAMPER_Enable</li> </ul>                                                                                                                                                                         |

### LL\_RTC\_TAMPER\_Disable

|                                                   |                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_TAMPER_Disable(<br/>    RTC_TypeDef * RTCx, uint32_t Tamper)</code>                                                                                                                                                                                                                                             |
| Function description                              | Clear RTC_TAMPx input detection.                                                                                                                                                                                                                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Tamper:</b> This parameter can be a combination of the following values: (*) value not defined in all devices.             <ul style="list-style-type: none"> <li>– LL_RTC_TAMPER_1</li> <li>– LL_RTC_TAMPER_2</li> <li>– LL_RTC_TAMPER_3 (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TAFCR TAMP1E LL_RTC_TAMPER_Disable</li> <li>TAFCR TAMP2E LL_RTC_TAMPER_Disable</li> </ul>                                                                                                                                                                                                                  |

- TAFCR TAMP3E LL\_RTC\_TAMPER\_Disable

### **LL\_RTC\_TAMPER\_DisablePullUp**

Function name      **\_\_STATIC\_INLINE void LL\_RTC\_TAMPER\_DisablePullUp( RTC\_TypeDef \* RTCx )**

Function description      Disable RTC\_TAMPx pull-up disable (Disable precharge of RTC\_TAMPx pins)

Parameters      • **RTCx:** RTC Instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• TAFCR TAMPPUDIS LL\_RTC\_TAMPER\_DisablePullUp

### **LL\_RTC\_TAMPER\_EnablePullUp**

Function name      **\_\_STATIC\_INLINE void LL\_RTC\_TAMPER\_EnablePullUp( RTC\_TypeDef \* RTCx )**

Function description      Enable RTC\_TAMPx pull-up disable ( Precharge RTC\_TAMPx pins before sampling)

Parameters      • **RTCx:** RTC Instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• TAFCR TAMPPUDIS LL\_RTC\_TAMPER\_EnablePullUp

### **LL\_RTC\_TAMPER\_SetPrecharge**

Function name      **\_\_STATIC\_INLINE void LL\_RTC\_TAMPER\_SetPrecharge( RTC\_TypeDef \* RTCx, uint32\_t Duration )**

Function description      Set RTC\_TAMPx precharge duration.

Parameters

- **RTCx:** RTC Instance
- **Duration:** This parameter can be one of the following values:
  - LL\_RTC\_TAMPER\_DURATION\_1RTCCLK
  - LL\_RTC\_TAMPER\_DURATION\_2RTCCLK
  - LL\_RTC\_TAMPER\_DURATION\_4RTCCLK
  - LL\_RTC\_TAMPER\_DURATION\_8RTCCLK

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• TAFCR TAMPPRCH LL\_RTC\_TAMPER\_SetPrecharge

### **LL\_RTC\_TAMPER\_GetPrecharge**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_RTC\_TAMPER\_GetPrecharge( RTC\_TypeDef \* RTCx )**

Function description      Get RTC\_TAMPx precharge duration.

|                                                   |                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                              |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_TAMPER_DURATION_1RTCCLK</li> <li>– LL_RTC_TAMPER_DURATION_2RTCCLK</li> <li>– LL_RTC_TAMPER_DURATION_4RTCCLK</li> <li>– LL_RTC_TAMPER_DURATION_8RTCCLK</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TAFCR TAMPPRCH LL_RTC_TAMPER_GetPrecharge</li> </ul>                                                                                                                                                                                                                                            |

### LL\_RTC\_TAMPER\_SetFilterCount

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_TAMPER_SetFilterCount( RTC_TypeDef * RTCx, uint32_t FilterCount )</code>                                                                                                                                                                                                                                                                      |
| Function description                              | Set RTC_TAMPx filter count.                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>FilterCount:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_TAMPER_FILTER_DISABLE</li> <li>– LL_RTC_TAMPER_FILTER_2SAMPLE</li> <li>– LL_RTC_TAMPER_FILTER_4SAMPLE</li> <li>– LL_RTC_TAMPER_FILTER_8SAMPLE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TAFCR TAMPFLT LL_RTC_TAMPER_SetFilterCount</li> </ul>                                                                                                                                                                                                                                                                                  |

### LL\_RTC\_TAMPER\_GetFilterCount

|                                                   |                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_TAMPER_GetFilterCount( RTC_TypeDef * RTCx )</code>                                                                                                                                                                                                                                         |
| Function description                              | Get RTC_TAMPx filter count.                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_TAMPER_FILTER_DISABLE</li> <li>– LL_RTC_TAMPER_FILTER_2SAMPLE</li> <li>– LL_RTC_TAMPER_FILTER_4SAMPLE</li> <li>– LL_RTC_TAMPER_FILTER_8SAMPLE</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TAFCR TAMPFLT LL_RTC_TAMPER_GetFilterCount</li> </ul>                                                                                                                                                                                                                                   |

### LL\_RTC\_TAMPER\_SetSamplingFreq

|                      |                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_TAMPER_SetSamplingFreq( RTC_TypeDef * RTCx, uint32_t SamplingFreq )</code> |
| Function description | Set Tamper sampling frequency.                                                                               |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>SamplingFreq:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_32768</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_16384</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_8192</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_4096</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_2048</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_1024</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_512</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_256</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TAFCR TAMPFREQ LL_RTC_TAMPER_SetSamplingFreq</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

### LL\_RTC\_TAMPER\_GetSamplingFreq

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_RTC_TAMPER_GetSamplingFreq (RTC_TypeDef * RTCx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description                              | Get Tamper sampling frequency.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_32768</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_16384</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_8192</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_4096</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_2048</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_1024</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_512</li> <li>- LL_RTC_TAMPER_SAMPLFREQDIV_256</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>TAFCR TAMPFREQ LL_RTC_TAMPER_GetSamplingFreq</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                           |

### LL\_RTC\_TAMPER\_EnableActiveLevel

|                      |                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_TAMPER_EnableActiveLevel<br/>(RTC_TypeDef * RTCx, uint32_t Tamper)</code>                                                                                                                                                                                                                                                                                     |
| Function description | Enable Active level for Tamper input.                                                                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>Tamper:</b> This parameter can be a combination of the following values: (*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- LL_RTC_TAMPER_ACTIVELEVEL_TAMP1</li> <li>- LL_RTC_TAMPER_ACTIVELEVEL_TAMP2</li> <li>- LL_RTC_TAMPER_ACTIVELEVEL_TAMP3 (*)</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                   |
| Reference Manual to  | <ul style="list-style-type: none"> <li>TAFCR TAMP1TRG LL_RTC_TAMPER_EnableActiveLevel</li> </ul>                                                                                                                                                                                                                                                                                                |

- LL API cross reference:
- TAFCR TAMP2TRG LL\_RTC\_TAMPER\_EnableActiveLevel
  - TAFCR TAMP3TRG LL\_RTC\_TAMPER\_EnableActiveLevel

### **LL\_RTC\_TAMPER\_DisableActiveLevel**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_RTC_TAMPER_DisableActiveLevel(RTC_TypeDef * RTCx, uint32_t Tamper)</code></b>                                                                                                                                                                                                                                                                             |
| Function description                        | Disable Active level for Tamper input.                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Tamper:</b> This parameter can be a combination of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- LL_RTC_TAMPER_ACTIVELEVEL_TAMP1</li> <li>- LL_RTC_TAMPER_ACTIVELEVEL_TAMP2</li> <li>- LL_RTC_TAMPER_ACTIVELEVEL_TAMP3 (*)</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                           |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• TAFCR TAMP1TRG LL_RTC_TAMPER_DisableActiveLevel</li> <li>• TAFCR TAMP2TRG LL_RTC_TAMPER_DisableActiveLevel</li> <li>• TAFCR TAMP3TRG LL_RTC_TAMPER_DisableActiveLevel</li> </ul>                                                                                                                                                                 |

### **LL\_RTC\_WAKEUP\_Enable**

|                                             |                                                                                                                                             |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_RTC_WAKEUP_Enable(RTC_TypeDef * RTCx)</code></b>                                                            |
| Function description                        | Enable Wakeup timer.                                                                                                                        |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                       | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR WUTE LL_RTC_WAKEUP_Enable</li> </ul>                                                            |

### **LL\_RTC\_WAKEUP\_Disable**

|                                             |                                                                                                                                             |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_RTC_WAKEUP_Disable(RTC_TypeDef * RTCx)</code></b>                                                           |
| Function description                        | Disable Wakeup timer.                                                                                                                       |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                       | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR WUTE LL_RTC_WAKEUP_Disable</li> </ul>                                                           |

**LL\_RTC\_WAKEUP\_IsEnabled**

Function name **`_STATIC_INLINE uint32_t LL_RTC_WAKEUP_IsEnabled(RTC_TypeDef * RTCx)`**

Function description Check if Wakeup timer is enabled or not.

Parameters • **RTCx:** RTC Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
• CR WUTE LL\_RTC\_WAKEUP\_IsEnabled

**LL\_RTC\_WAKEUP\_SetClock**

Function name **`_STATIC_INLINE void LL_RTC_WAKEUP_SetClock(RTC_TypeDef * RTCx, uint32_t WakeupClock)`**

Function description Select Wakeup clock.

Parameters

- **RTCx:** RTC Instance
- **WakeupClock:** This parameter can be one of the following values:
  - LL\_RTC\_WAKEUPCLOCK\_DIV\_16
  - LL\_RTC\_WAKEUPCLOCK\_DIV\_8
  - LL\_RTC\_WAKEUPCLOCK\_DIV\_4
  - LL\_RTC\_WAKEUPCLOCK\_DIV\_2
  - LL\_RTC\_WAKEUPCLOCK\_CKSPRE
  - LL\_RTC\_WAKEUPCLOCK\_CKSPRE\_WUT

Return values • **None**

Notes

- Bit is write-protected. LL\_RTC\_DisableWriteProtection function should be called before.
- Bit can be written only when RTC\_CR WUTE bit = 0 and RTC\_ISR WUTWF bit = 1

Reference Manual to  
LL API cross  
reference:  
• CR WUCKSEL LL\_RTC\_WAKEUP\_SetClock

**LL\_RTC\_WAKEUP\_GetClock**

Function name **`_STATIC_INLINE uint32_t LL_RTC_WAKEUP_GetClock(RTC_TypeDef * RTCx)`**

Function description Get Wakeup clock.

Parameters

- **RTCx:** RTC Instance

Return values

- **Returned:** value can be one of the following values:
  - LL\_RTC\_WAKEUPCLOCK\_DIV\_16
  - LL\_RTC\_WAKEUPCLOCK\_DIV\_8
  - LL\_RTC\_WAKEUPCLOCK\_DIV\_4
  - LL\_RTC\_WAKEUPCLOCK\_DIV\_2
  - LL\_RTC\_WAKEUPCLOCK\_CKSPRE
  - LL\_RTC\_WAKEUPCLOCK\_CKSPRE\_WUT

- Reference Manual to  
LL API cross  
reference:
- CR WUCKSEL LL\_RTC\_WAKEUP\_GetClock

### **LL\_RTC\_WAKEUP\_SetAutoReload**

|                                                   |                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_RTC_WAKEUP_SetAutoReload(RTC_TypeDef * RTCx, uint32_t Value)</code></b>                                                |
| Function description                              | Set Wakeup auto-reload value.                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Value:</b> Value between Min_Data=0x00 and Max_Data=0xFFFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit can be written only when WUTWF is set to 1 in RTC_ISR</li> </ul>                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• WUTR WUT LL_RTC_WAKEUP_SetAutoReload</li> </ul>                                                               |

### **LL\_RTC\_WAKEUP\_GetAutoReload**

|                                                   |                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_RTC_WAKEUP_GetAutoReload(RTC_TypeDef * RTCx)</code></b>                 |
| Function description                              | Get Wakeup auto-reload value.                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xFFFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• WUTR WUT LL_RTC_WAKEUP_GetAutoReload</li> </ul>                    |

### **LL\_RTC\_BAK\_SetRegister**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_RTC_BAK_SetRegister(RTC_TypeDef * RTCx, uint32_t BackupRegister, uint32_t Data)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Writes a data in a specified RTC Backup data register.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>BackupRegister:</b> This parameter can be one of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- LL_RTC_BKP_DR0</li> <li>- LL_RTC_BKP_DR1</li> <li>- LL_RTC_BKP_DR2</li> <li>- LL_RTC_BKP_DR3</li> <li>- LL_RTC_BKP_DR4</li> <li>- LL_RTC_BKP_DR5 (*)</li> <li>- LL_RTC_BKP_DR6 (*)</li> <li>- LL_RTC_BKP_DR7 (*)</li> <li>- LL_RTC_BKP_DR8 (*)</li> <li>- LL_RTC_BKP_DR9 (*)</li> </ul> </li> </ul> |

- LL\_RTC\_BKP\_DR10 (\*)
- LL\_RTC\_BKP\_DR11 (\*)
- LL\_RTC\_BKP\_DR12 (\*)
- LL\_RTC\_BKP\_DR13 (\*)
- LL\_RTC\_BKP\_DR14 (\*)
- LL\_RTC\_BKP\_DR15 (\*)
- LL\_RTC\_BKP\_DR16 (\*)
- LL\_RTC\_BKP\_DR17 (\*)
- LL\_RTC\_BKP\_DR18 (\*)
- LL\_RTC\_BKP\_DR19 (\*)
- LL\_RTC\_BKP\_DR20 (\*)
- LL\_RTC\_BKP\_DR21 (\*)
- LL\_RTC\_BKP\_DR22 (\*)
- LL\_RTC\_BKP\_DR23 (\*)
- LL\_RTC\_BKP\_DR24 (\*)
- LL\_RTC\_BKP\_DR25 (\*)
- LL\_RTC\_BKP\_DR26 (\*)
- LL\_RTC\_BKP\_DR27 (\*)
- LL\_RTC\_BKP\_DR28 (\*)
- LL\_RTC\_BKP\_DR29 (\*)
- LL\_RTC\_BKP\_DR30 (\*)
- LL\_RTC\_BKP\_DR31 (\*)

- **Data:** Value between Min\_Data=0x00 and Max\_Data=0xFFFFFFFF

#### Return values

Reference Manual to  
LL API cross  
reference:

- **None**

- BKPxR BKP LL\_RTC\_BAK\_SetRegister

## LL\_RTC\_BAK\_GetRegister

Function name **\_STATIC\_INLINE uint32\_t LL\_RTC\_BAK\_GetRegister(RTC\_TypeDef \* RTCx, uint32\_t BackupRegister)**

Function description Reads data from the specified RTC Backup data Register.

- |            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>BackupRegister:</b> This parameter can be one of the following values: (*) value not defined in all devices.</li> </ul> <ul style="list-style-type: none"> <li>- LL_RTC_BKP_DR0</li> <li>- LL_RTC_BKP_DR1</li> <li>- LL_RTC_BKP_DR2</li> <li>- LL_RTC_BKP_DR3</li> <li>- LL_RTC_BKP_DR4</li> <li>- LL_RTC_BKP_DR5 (*)</li> <li>- LL_RTC_BKP_DR6 (*)</li> <li>- LL_RTC_BKP_DR7 (*)</li> <li>- LL_RTC_BKP_DR8 (*)</li> <li>- LL_RTC_BKP_DR9 (*)</li> <li>- LL_RTC_BKP_DR10 (*)</li> <li>- LL_RTC_BKP_DR11 (*)</li> <li>- LL_RTC_BKP_DR12 (*)</li> <li>- LL_RTC_BKP_DR13 (*)</li> </ul> |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

- LL\_RTC\_BKP\_DR14 (\*)
- LL\_RTC\_BKP\_DR15 (\*)
- LL\_RTC\_BKP\_DR16 (\*)
- LL\_RTC\_BKP\_DR17 (\*)
- LL\_RTC\_BKP\_DR18 (\*)
- LL\_RTC\_BKP\_DR19 (\*)
- LL\_RTC\_BKP\_DR20 (\*)
- LL\_RTC\_BKP\_DR21 (\*)
- LL\_RTC\_BKP\_DR22 (\*)
- LL\_RTC\_BKP\_DR23 (\*)
- LL\_RTC\_BKP\_DR24 (\*)
- LL\_RTC\_BKP\_DR25 (\*)
- LL\_RTC\_BKP\_DR26 (\*)
- LL\_RTC\_BKP\_DR27 (\*)
- LL\_RTC\_BKP\_DR28 (\*)
- LL\_RTC\_BKP\_DR29 (\*)
- LL\_RTC\_BKP\_DR30 (\*)
- LL\_RTC\_BKP\_DR31 (\*)

|                                                   |                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xFFFFFFFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BKPxR BKP LL_RTC_BAK_GetRegister</li> </ul>                            |

### LL\_RTC\_CAL\_SetOutputFreq

|                                                   |                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_CAL_SetOutputFreq(<br/>  RTC_TypeDef * RTCx, uint32_t Frequency)</code>                                                                                                                                                                                                                     |
| Function description                              | Set Calibration output frequency (1 Hz or 512 Hz)                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Frequency:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_CALIB_OUTPUT_NONE</li> <li>- LL_RTC_CALIB_OUTPUT_1HZ</li> <li>- LL_RTC_CALIB_OUTPUT_512HZ</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bits are write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul>                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR COE LL_RTC_CAL_SetOutputFreq</li> <li>• CR COSEL LL_RTC_CAL_SetOutputFreq</li> </ul>                                                                                                                                                                                              |

### LL\_RTC\_CAL\_GetOutputFreq

|                      |                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_RTC_CAL_GetOutputFreq(<br/>  RTC_TypeDef * RTCx)</code> |
| Function description | Get Calibration output frequency (1 Hz or 512 Hz)                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>             |

|                                                   |                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_CALIB_OUTPUT_NONE</li> <li>- LL_RTC_CALIB_OUTPUT_1HZ</li> <li>- LL_RTC_CALIB_OUTPUT_512HZ</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR COE LL_RTC_CAL_SetOutputFreq</li> <li>• CR COSEL LL_RTC_CAL_SetOutputFreq</li> </ul>                                                                                                                                             |

### LL\_RTC\_CAL\_SetPulse

|                                                   |                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_CAL_SetPulse (RTC_TypeDef * RTCx, uint32_t Pulse)</code>                                                                                                                                                                                                     |
| Function description                              | Insert or not One RTCCLK pulse every 2exp11 pulses (frequency increased by 488.5 ppm)                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Pulse:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_CALIB_INSERTPULSE_NONE</li> <li>- LL_RTC_CALIB_INSERTPULSE_SET</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> <li>• Bit can be written only when RECALPF is set to 0 in RTC_ISR</li> </ul>                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CALR CALP LL_RTC_CAL_SetPulse</li> </ul>                                                                                                                                                                                                              |

### LL\_RTC\_CAL\_IsPulseInserted

|                                                   |                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_CAL_IsPulseInserted (RTC_TypeDef * RTCx)</code>               |
| Function description                              | Check if one RTCCLK has been inserted or not every 2exp11 pulses (frequency increased by 488.5 ppm) |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CALR CALP LL_RTC_CAL_IsPulseInserted</li> </ul>            |

### LL\_RTC\_CAL\_SetPeriod

|                      |                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_CAL_SetPeriod (RTC_TypeDef * RTCx, uint32_t Period)</code>                                                                                                                                                                                             |
| Function description | Set the calibration cycle period.                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>Period:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_RTC_CALIB_PERIOD_32SEC</li> <li>- LL_RTC_CALIB_PERIOD_16SEC</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | – LL_RTC_CALIB_PERIOD_8SEC                                                                                                                                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> <li>• Bit can be written only when RECALPF is set to 0 in RTC_ISR</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CALR CALW8 LL_RTC_CAL_SetPeriod</li> <li>• CALR CALW16 LL_RTC_CAL_SetPeriod</li> </ul>                                                                                    |

### LL\_RTC\_CAL\_GetPeriod

|                                                   |                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE uint32_t LL_RTC_CAL_GetPeriod<br/>(RTC_TypeDef * RTCx)</b>                                                                                                                                                                                                     |
| Function description                              | Get the calibration cycle period.                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_RTC_CALIB_PERIOD_32SEC</li> <li>– LL_RTC_CALIB_PERIOD_16SEC</li> <li>– LL_RTC_CALIB_PERIOD_8SEC</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CALR CALW8 LL_RTC_CAL_GetPeriod</li> <li>• CALR CALW16 LL_RTC_CAL_GetPeriod</li> </ul>                                                                                                                                                  |

### LL\_RTC\_CAL\_SetMinus

|                                                   |                                                                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_RTC_CAL_SetMinus<br/>(RTC_TypeDef * RTCx, uint32_t CalibMinus)</b>                                                                                                                       |
| Function description                              | Set Calibration minus.                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>CalibMinus:</b> Value between Min_Data=0x00 and Max_Data=0x1FF</li> </ul>                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> <li>• Bit can be written only when RECALPF is set to 0 in RTC_ISR</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CALR CALM LL_RTC_CAL_SetMinus</li> </ul>                                                                                                                                  |

### LL\_RTC\_CAL\_GetMinus

|                      |                                                                             |
|----------------------|-----------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE uint32_t LL_RTC_CAL_GetMinus<br/>(RTC_TypeDef * RTCx)</b> |
| Function description | Get Calibration minus.                                                      |

---

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                               |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Value:</b> between Min_Data=0x00 and Max_Data= 0x1FF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CALR CALM LL_RTC_CAL_GetMinus</li> </ul>                           |

### LL\_RTC\_IsActiveFlag\_RECALP

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_RECALP<br/>(RTC_TypeDef * RTCx)</code> |
| Function description                              | Get Recalibration pending Flag.                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR RECALPF LL_RTC_IsActiveFlag_RECALP</li> </ul>  |

### LL\_RTC\_IsActiveFlag\_TAMP3

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_TAMP3<br/>(RTC_TypeDef * RTCx)</code> |
| Function description                              | Get RTC_TAMP3 detection flag.                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR TAMP3F LL_RTC_IsActiveFlag_TAMP3</li> </ul>   |

### LL\_RTC\_IsActiveFlag\_TAMP2

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_TAMP2<br/>(RTC_TypeDef * RTCx)</code> |
| Function description                              | Get RTC_TAMP2 detection flag.                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR TAMP2F LL_RTC_IsActiveFlag_TAMP2</li> </ul>   |

### LL\_RTC\_IsActiveFlag\_TAMP1

|                      |                                                                                          |
|----------------------|------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_TAMP1<br/>(RTC_TypeDef * RTCx)</code> |
| Function description | Get RTC_TAMP1 detection flag.                                                            |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>              |

---

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR TAMP1F LL_RTC_IsActiveFlag_TAMP1</li> </ul> |

### LL\_RTC\_IsActiveFlag\_TSOV

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_TSOV(<br/>(RTC_TypeDef * RTCx))</code> |
| Function description                              | Get Time-stamp overflow flag.                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR TSOVF LL_RTC_IsActiveFlag_TSOV</li> </ul>      |

### LL\_RTC\_IsActiveFlag\_TS

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_TS(<br/>(RTC_TypeDef * RTCx))</code> |
| Function description                              | Get Time-stamp flag.                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR TSF LL_RTC_IsActiveFlag_TS</li> </ul>        |

### LL\_RTC\_IsActiveFlag\_WUT

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_WUT(<br/>(RTC_TypeDef * RTCx))</code> |
| Function description                              | Get Wakeup timer flag.                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>ISR WUTF LL_RTC_IsActiveFlag_WUT</li> </ul>       |

### LL\_RTC\_IsActiveFlag\_ALRB

|                      |                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_ALRB(<br/>(RTC_TypeDef * RTCx))</code> |
| Function description | Get Alarm B flag.                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>               |
| Return values        | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>          |

- Reference Manual to  
LL API cross  
reference:
- ISR ALRBF LL\_RTC\_IsActiveFlag\_ALRB

### **LL\_RTC\_IsActiveFlag\_ALRA**

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_ALRA<br/>(RTC_TypeDef * RTCx)</code> |
| Function description | Get Alarm A flag.                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>      |

Reference Manual to  
LL API cross  
reference:

- ISR ALRAF LL\_RTC\_IsActiveFlag\_ALRA

### **LL\_RTC\_ClearFlag\_TAMP3**

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_ClearFlag_TAMP3<br/>(RTC_TypeDef * RTCx)</code> |
| Function description | Clear RTC_TAMP3 detection flag.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                   |

Reference Manual to  
LL API cross  
reference:

- ISR TAMP3F LL\_RTC\_ClearFlag\_TAMP3

### **LL\_RTC\_ClearFlag\_TAMP2**

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_ClearFlag_TAMP2<br/>(RTC_TypeDef * RTCx)</code> |
| Function description | Clear RTC_TAMP2 detection flag.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                   |

Reference Manual to  
LL API cross  
reference:

- ISR TAMP2F LL\_RTC\_ClearFlag\_TAMP2

### **LL\_RTC\_ClearFlag\_TAMP1**

|                      |                                                                                   |
|----------------------|-----------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_ClearFlag_TAMP1<br/>(RTC_TypeDef * RTCx)</code> |
| Function description | Clear RTC_TAMP1 detection flag.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                   |

Reference Manual to  
LL API cross

- ISR TAMP1F LL\_RTC\_ClearFlag\_TAMP1

reference:

### LL\_RTC\_ClearFlag\_TSOV

Function name **`__STATIC_INLINE void LL_RTC_ClearFlag_TSOV(RTC_TypeDef * RTCx)`**

Function description Clear Time-stamp overflow flag.

Parameters • **RTCx:** RTC Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
[TSOVF LL\\_RTC\\_ClearFlag\\_TSOV](#)

### LL\_RTC\_ClearFlag\_TS

Function name **`__STATIC_INLINE void LL_RTC_ClearFlag_TS (RTC_TypeDef * RTCx)`**

Function description Clear Time-stamp flag.

Parameters • **RTCx:** RTC Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
[TSF LL\\_RTC\\_ClearFlag\\_TS](#)

### LL\_RTC\_ClearFlag\_WUT

Function name **`__STATIC_INLINE void LL_RTC_ClearFlag_WUT( RTC_TypeDef * RTCx )`**

Function description Clear Wakeup timer flag.

Parameters • **RTCx:** RTC Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
[WUTF LL\\_RTC\\_ClearFlag\\_WUT](#)

### LL\_RTC\_ClearFlag\_ALRB

Function name **`__STATIC_INLINE void LL_RTC_ClearFlag_ALRB( RTC_TypeDef * RTCx )`**

Function description Clear Alarm B flag.

Parameters • **RTCx:** RTC Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
[ALRBF LL\\_RTC\\_ClearFlag\\_ALRB](#)

**LL\_RTC\_ClearFlag\_ALRA**

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_ClearFlag_ALRA<br/>(RTC_TypeDef * RTCx)</code>  |
| Function description                              | Clear Alarm A flag.                                                               |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>RTCx:</b> RTC Instance</li></ul>       |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• ISR ALRAF LL_RTC_ClearFlag_ALRA</li></ul> |

**LL\_RTC\_IsActiveFlag\_INIT**

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_INIT<br/>(RTC_TypeDef * RTCx)</code> |
| Function description                              | Get Initialization flag.                                                                |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>RTCx:</b> RTC Instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• ISR INITF LL_RTC_IsActiveFlag_INIT</li></ul>    |

**LL\_RTC\_IsActiveFlag\_RS**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_RS<br/>(RTC_TypeDef * RTCx)</code> |
| Function description                              | Get Registers synchronization flag.                                                   |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>RTCx:</b> RTC Instance</li></ul>           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• ISR RSF LL_RTC_IsActiveFlag_RS</li></ul>      |

**LL\_RTC\_ClearFlag\_RS**

|                                                   |                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_ClearFlag_RS (RTC_TypeDef<br/>* RTCx)</code> |
| Function description                              | Clear Registers synchronization flag.                                          |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>RTCx:</b> RTC Instance</li></ul>    |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• ISR RSF LL_RTC_ClearFlag_RS</li></ul>  |

**LL\_RTC\_IsActiveFlag\_INITS**

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_INITS(<br/>    RTC_TypeDef * RTCx)</code> |
| Function description                              | Get Initialization status flag.                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR INITS LL_RTC_IsActiveFlag_INITS</li> </ul>      |

**LL\_RTC\_IsActiveFlag\_SHP**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_SHP(<br/>    RTC_TypeDef * RTCx)</code> |
| Function description                              | Get Shift operation pending flag.                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR SHPF LL_RTC_IsActiveFlag_SHP</li> </ul>       |

**LL\_RTC\_IsActiveFlag\_WUTW**

|                                                   |                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_WUTW(<br/>    RTC_TypeDef * RTCx)</code> |
| Function description                              | Get Wakeup timer write flag.                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR WUTWF LL_RTC_IsActiveFlag_WUTW</li> </ul>      |

**LL\_RTC\_IsActiveFlag\_ALRBW**

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_ALRBW(<br/>    RTC_TypeDef * RTCx)</code> |
| Function description                              | Get Alarm B write flag.                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR ALRBWF LL_RTC_IsActiveFlag_ALRBW</li> </ul>     |

**LL\_RTC\_IsActiveFlag\_ALRAW**

|                                                   |                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_RTC_IsActiveFlag_ALRAW(RTC_TypeDef * RTCx)</code>      |
| Function description                              | Get Alarm A write flag.                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR ALRAWF LL_RTC_IsActiveFlag_ALRAW</li> </ul> |

**LL\_RTC\_EnableIT\_TS**

|                                                   |                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_EnableIT_TS(RTC_TypeDef * RTCx)</code>                                                                    |
| Function description                              | Enable Time-stamp interrupt.                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR TSIE LL_RTC_EnableIT_TS</li> </ul>                                                              |

**LL\_RTC\_DisableIT\_TS**

|                                                   |                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_RTC_DisableIT_TS(RTC_TypeDef * RTCx)</code>                                                                   |
| Function description                              | Disable Time-stamp interrupt.                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR TSIE LL_RTC_DisableIT_TS</li> </ul>                                                             |

**LL\_RTC\_EnableIT\_WUT**

|                      |                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_RTC_EnableIT_WUT(RTC_TypeDef * RTCx)</code>                                 |
| Function description | Enable Wakeup timer interrupt.                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                           |
| Notes                | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection</li> </ul> |

function should be called before.

Reference Manual to  
LL API cross  
reference:

- CR WUTIE LL\_RTC\_EnableIT\_WUT

### LL\_RTC\_DisableIT\_WUT

Function name

**`__STATIC_INLINE void LL_RTC_DisableIT_WUT  
(RTC_TypeDef * RTCx)`**

Function description

Disable Wakeup timer interrupt.

Parameters

- **RTCx:** RTC Instance

Return values

- **None**

Notes

- Bit is write-protected. LL\_RTC\_DisableWriteProtection function should be called before.

Reference Manual to  
LL API cross  
reference:

- CR WUTIE LL\_RTC\_DisableIT\_WUT

### LL\_RTC\_EnableIT\_ALRB

Function name

**`__STATIC_INLINE void LL_RTC_EnableIT_ALRB  
(RTC_TypeDef * RTCx)`**

Function description

Enable Alarm B interrupt.

Parameters

- **RTCx:** RTC Instance

Return values

- **None**

Notes

- Bit is write-protected. LL\_RTC\_DisableWriteProtection function should be called before.

Reference Manual to  
LL API cross  
reference:

- CR ALRBIE LL\_RTC\_EnableIT\_ALRB

### LL\_RTC\_DisableIT\_ALRB

Function name

**`__STATIC_INLINE void LL_RTC_DisableIT_ALRB  
(RTC_TypeDef * RTCx)`**

Function description

Disable Alarm B interrupt.

Parameters

- **RTCx:** RTC Instance

Return values

- **None**

Notes

- Bit is write-protected. LL\_RTC\_DisableWriteProtection function should be called before.

Reference Manual to  
LL API cross  
reference:

- CR ALRBIE LL\_RTC\_DisableIT\_ALRB

**LL\_RTC\_EnableIT\_ALRA**

|                                                   |                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_EnableIT_ALRA<br/>(RTC_TypeDef * RTCx)</code>                                                              |
| Function description                              | Enable Alarm A interrupt.                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR ALRAIE LL_RTC_EnableIT_ALRA</li> </ul>                                                          |

**LL\_RTC\_DisableIT\_ALRA**

|                                                   |                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_DisableIT_ALRA<br/>(RTC_TypeDef * RTCx)</code>                                                             |
| Function description                              | Disable Alarm A interrupt.                                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Bit is write-protected. LL_RTC_DisableWriteProtection function should be called before.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR ALRAIE LL_RTC_DisableIT_ALRA</li> </ul>                                                         |

**LL\_RTC\_EnableIT\_TAMP**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_RTC_EnableIT_TAMP<br/>(RTC_TypeDef * RTCx)</code>        |
| Function description                              | Enable all Tamper Interrupt.                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TAFCR TAMPIE LL_RTC_EnableIT_TAMP</li> </ul> |

**LL\_RTC\_DisableIT\_TAMP**

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_RTC_DisableIT_TAMP<br/>(RTC_TypeDef * RTCx)</code>        |
| Function description | Disable all Tamper Interrupt.                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                        |
| Reference Manual to  | <ul style="list-style-type: none"> <li>• TAFCR TAMPIE LL_RTC_DisableIT_TAMP</li> </ul> |

LL API cross  
reference:

### **LL\_RTC\_IsEnabledIT\_TS**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_RTC\_IsEnabledIT\_TS  
(RTC\_TypeDef \* RTCx)**

Function description      Check if Time-stamp interrupt is enabled or not.

Parameters      • **RTCx:** RTC Instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

### **LL\_RTC\_IsEnabledIT\_WUT**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_RTC\_IsEnabledIT\_WUT  
(RTC\_TypeDef \* RTCx)**

Function description      Check if Wakeup timer interrupt is enabled or not.

Parameters      • **RTCx:** RTC Instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

### **LL\_RTC\_IsEnabledIT\_ALRB**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_RTC\_IsEnabledIT\_ALRB  
(RTC\_TypeDef \* RTCx)**

Function description      Check if Alarm B interrupt is enabled or not.

Parameters      • **RTCx:** RTC Instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

### **LL\_RTC\_IsEnabledIT\_ALRA**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_RTC\_IsEnabledIT\_ALRA  
(RTC\_TypeDef \* RTCx)**

Function description      Check if Alarm A interrupt is enabled or not.

Parameters      • **RTCx:** RTC Instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross

reference:

### **LL\_RTC\_IsEnabledIT\_TAMP**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_RTC_IsEnabledIT_TAMP( RTC_TypeDef * RTCx )</code></b> |
| Function description                              | Check if all the TAMPER interrupts are enabled or not.                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | • TAFCR TAMPIE LL_RTC_IsEnabledIT_TAMP                                                     |

### **LL\_RTC\_DeInit**

|                      |                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>ErrorStatus LL_RTC_DeInit ( RTC_TypeDef * RTCx )</code></b>                                                                                                                                                                                          |
| Function description | De-Initializes the RTC registers to their default reset values.                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                 |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: RTC registers are de-initialized</li> <li>– ERROR: RTC registers are not de-initialized</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• This function doesn't reset the RTC Clock source and RTC Backup Data registers.</li> </ul>                                                                                                                           |

### **LL\_RTC\_Init**

|                      |                                                                                                                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>ErrorStatus LL_RTC_Init ( RTC_TypeDef * RTCx, LL_RTC_InitTypeDef * RTC_InitStruct )</code></b>                                                                                                                                                 |
| Function description | Initializes the RTC registers according to the specified parameters in RTC_InitStruct.                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>RTC_InitStruct:</b> pointer to a LL_RTC_InitTypeDef structure that contains the configuration information for the RTC peripheral.</li> </ul>                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: RTC registers are initialized</li> <li>– ERROR: RTC registers are not initialized</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• The RTC Prescaler register is write protected and can be written in initialization mode only.</li> </ul>                                                                                                       |

**LL\_RTC\_StructInit**

|                      |                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_RTC_StructInit (LL_RTC_InitTypeDef * RTC_InitStruct)</b>                                                                             |
| Function description | Set each LL_RTC_InitTypeDef field to default value.                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTC_InitStruct:</b> pointer to a LL_RTC_InitTypeDef structure which will be initialized.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                 |

**LL\_RTC\_TIME\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_RTC_TIME_Init (RTC_TypeDef * RTCx, uint32_t RTC_Format, LL_RTC_TimeTypeDef * RTC_TimeStruct)</b>                                                                                                                                                                                                                                                                                        |
| Function description | Set the RTC current time.                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>RTC_Format:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_RTC_FORMAT_BIN</li> <li>– LL_RTC_FORMAT_BCD</li> </ul> </li> <li>• <b>RTC_TimeStruct:</b> pointer to a RTC_TimeTypeDef structure that contains the time configuration information for the RTC.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: RTC Time register is configured</li> <li>– ERROR: RTC Time register is not configured</li> </ul> </li> </ul>                                                                                                                                                         |

**LL\_RTC\_TIME\_StructInit**

|                      |                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_RTC_TIME_StructInit (LL_RTC_TimeTypeDef * RTC_TimeStruct)</b>                                                                        |
| Function description | Set each LL_RTC_TimeTypeDef field to default value (Time = 00h:00min:00sec).                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTC_TimeStruct:</b> pointer to a LL_RTC_TimeTypeDef structure which will be initialized.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                 |

**LL\_RTC\_DATE\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_RTC_DATE_Init (RTC_TypeDef * RTCx, uint32_t RTC_Format, LL_RTC_DateTypeDef * RTC_DateStruct)</b>                                                                                                                                                                                                                          |
| Function description | Set the RTC current date.                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> <li>• <b>RTC_Format:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_RTC_FORMAT_BIN</li> <li>– LL_RTC_FORMAT_BCD</li> </ul> </li> <li>• <b>RTC_DateStruct:</b> pointer to a RTC_DateTypeDef structure</li> </ul> |

that contains the date configuration information for the RTC.

|               |                                                                                                                                                                                                                                                     |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>SUCCESS: RTC Day register is configured</li> <li>ERROR: RTC Day register is not configured</li> </ul> </li> </ul> |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### LL\_RTC\_DATE\_StructInit

|                      |                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void LL_RTC_DATE_StructInit (LL_RTC_DateTypeDef * RTC_DateStruct)</code>                                                                |
| Function description | Set each LL_RTC_DateTypeDef field to default value (date = Monday, January 01 xx00)                                                           |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTC_DateStruct:</b> pointer to a LL_RTC_DateTypeDef structure which will be initialized.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                 |

### LL\_RTC\_ALMA\_Init

|                      |                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_RTC_ALMA_Init (RTC_TypeDef * RTCx, uint32_t RTC_Format, LL_RTC_AlarmTypeDef * RTC_AlarmStruct)</code>                                                                                                                                                                                                                                                                         |
| Function description | Set the RTC Alarm A.                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>RTC_Format:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>LL_RTC_FORMAT_BIN</li> <li>LL_RTC_FORMAT_BCD</li> </ul> </li> <li><b>RTC_AlarmStruct:</b> pointer to a LL_RTC_AlarmTypeDef structure that contains the alarm configuration parameters.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>SUCCESS: ALARMA registers are configured</li> <li>ERROR: ALARMA registers are not configured</li> </ul> </li> </ul>                                                                                                                                              |

**Notes**

- The Alarm register can only be written when the corresponding Alarm is disabled (Use LL\_RTC\_ALMA\_Disable function).

### LL\_RTC\_ALMB\_Init

|                      |                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_RTC_ALMB_Init (RTC_TypeDef * RTCx, uint32_t RTC_Format, LL_RTC_AlarmTypeDef * RTC_AlarmStruct)</code>                                                                                                                                                                                                                                                                         |
| Function description | Set the RTC Alarm B.                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> <li><b>RTC_Format:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>LL_RTC_FORMAT_BIN</li> <li>LL_RTC_FORMAT_BCD</li> </ul> </li> <li><b>RTC_AlarmStruct:</b> pointer to a LL_RTC_AlarmTypeDef structure that contains the alarm configuration parameters.</li> </ul> |

|               |                                                                                                                                                                                                                                                       |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>SUCCESS: ALARMB registers are configured</li> <li>ERROR: ALARMB registers are not configured</li> </ul> </li> </ul> |
| Notes         | <ul style="list-style-type: none"> <li>The Alarm register can only be written when the corresponding Alarm is disabled (LL_RTC_ALMB_Disable function).</li> </ul>                                                                                     |

### LL\_RTC\_ALMA\_StructInit

|                      |                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_RTC_ALMA_StructInit (LL_RTC_AlarmTypeDef * RTC_AlarmStruct)</b>                                                                       |
| Function description | Set each LL_RTC_AlarmTypeDef of ALARMA field to default value (Time = 00h:00mn:00sec / Day = 1st day of the month/Mask = all fields are masked). |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTC_AlarmStruct:</b> pointer to a LL_RTC_AlarmTypeDef structure which will be initialized.</li> </ul>  |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                    |

### LL\_RTC\_ALMB\_StructInit

|                      |                                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_RTC_ALMB_StructInit (LL_RTC_AlarmTypeDef * RTC_AlarmStruct)</b>                                                                       |
| Function description | Set each LL_RTC_AlarmTypeDef of ALARMA field to default value (Time = 00h:00mn:00sec / Day = 1st day of the month/Mask = all fields are masked). |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTC_AlarmStruct:</b> pointer to a LL_RTC_AlarmTypeDef structure which will be initialized.</li> </ul>  |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                    |

### LL\_RTC\_EnterInitMode

|                      |                                                                                                                                                                                                                               |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_RTC_EnterInitMode (RTC_TypeDef * RTCx)</b>                                                                                                                                                                  |
| Function description | Enters the RTC Initialization mode.                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                   |
| Return values        | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>SUCCESS: RTC is in Init mode</li> <li>ERROR: RTC is not in Init mode</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>The RTC Initialization mode is write protected, use the LL_RTC_DisableWriteProtection before calling this function.</li> </ul>                                                         |

### LL\_RTC\_ExitInitMode

|                      |                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_RTC_ExitInitMode (RTC_TypeDef * RTCx)</b>                                 |
| Function description | Exit the RTC Initialization mode.                                                           |
| Parameters           | <ul style="list-style-type: none"> <li><b>RTCx:</b> RTC Instance</li> </ul>                 |
| Return values        | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:</li> </ul> |

---

|       |                                                                                                                                                                                                                                                                                        |
|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       | <ul style="list-style-type: none"> <li>– SUCCESS: RTC exited from in Init mode</li> <li>– ERROR: Not applicable</li> </ul>                                                                                                                                                             |
| Notes | <ul style="list-style-type: none"> <li>• When the initialization sequence is complete, the calendar restarts counting after 4 RTCCLK cycles.</li> <li>• The RTC Initialization mode is write protected, use the LL_RTC_DisableWriteProtection before calling this function.</li> </ul> |

### LL\_RTC\_WaitForSynchro

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_RTC_WaitForSynchro (RTC_TypeDef * RTCx)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Waits until the RTC Time and Day registers (RTC_TR and RTC_DR) are synchronized with RTC APB clock.                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>RTCx:</b> RTC Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: RTC registers are synchronised</li> <li>– ERROR: RTC registers are not synchronised</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                              |
| Notes                | <ul style="list-style-type: none"> <li>• The RTC Resynchronization mode is write protected, use the LL_RTC_DisableWriteProtection before calling this function.</li> <li>• To read the calendar through the shadow registers after Calendar initialization, calendar update or after wakeup from low power modes the software must first clear the RSF flag. The software must then wait until it is set again before reading the calendar, which means that the calendar registers have been correctly copied into the RTC_TR and RTC_DR shadow registers.</li> </ul> |

## 75.3 RTC Firmware driver defines

### 75.3.1 RTC

#### ALARM OUTPUT

|                         |                        |
|-------------------------|------------------------|
| LL_RTC_ALARMOUT_DISABLE | Output disabled        |
| LL_RTC_ALARMOUT_ALMA    | Alarm A output enabled |
| LL_RTC_ALARMOUT_ALMB    | Alarm B output enabled |
| LL_RTC_ALARMOUT_WAKEUP  | Wakeups output enabled |

#### ALARM OUTPUT TYPE

|                                   |                                                      |
|-----------------------------------|------------------------------------------------------|
| LL_RTC_ALARM_OUTPUTTYPE_OPENDRAIN | RTC_ALARM, when mapped on PC13, is open-drain output |
| LL_RTC_ALARM_OUTPUTTYPE_PUSHPULL  | RTC_ALARM, when mapped on PC13, is push-pull output  |

#### ALARMA MASK

|                              |                                            |
|------------------------------|--------------------------------------------|
| LL_RTC_ALMA_MASK_NONE        | No masks applied on Alarm A                |
| LL_RTC_ALMA_MASK_DATEWEEKDAY | Date/day do not care in Alarm A comparison |
| LL_RTC_ALMA_MASK_HOURS       | Hours do not care in Alarm A comparison    |
| LL_RTC_ALMA_MASK_MINUTES     | Minutes do not care in Alarm A comparison  |

|                          |                                           |
|--------------------------|-------------------------------------------|
| LL_RTC_ALMA_MASK_SECONDS | Seconds do not care in Alarm A comparison |
| LL_RTC_ALMA_MASK_ALL     | Masks all                                 |

**ALARMA TIME FORMAT**

LL\_RTC\_ALMA\_TIME\_FORMAT\_AM AM or 24-hour format

LL\_RTC\_ALMA\_TIME\_FORMAT\_PM PM

**RTC Alarm A Date WeekDay**

LL\_RTC\_ALMA\_DATEWEEKDAYSEL\_DATE Alarm A Date is selected

LL\_RTC\_ALMA\_DATEWEEKDAYSEL\_WEEKDAY Alarm A WeekDay is selected

**ALARMB MASK**

LL\_RTC\_ALMB\_MASK\_NONE No masks applied on Alarm B

LL\_RTC\_ALMB\_MASK\_DATEWEEKDAY Date/day do not care in Alarm B comparison

LL\_RTC\_ALMB\_MASK\_HOURS Hours do not care in Alarm B comparison

LL\_RTC\_ALMB\_MASK\_MINUTES Minutes do not care in Alarm B comparison

LL\_RTC\_ALMB\_MASK\_SECONDS Seconds do not care in Alarm B comparison

LL\_RTC\_ALMB\_MASK\_ALL Masks all

**ALARMB TIME FORMAT**

LL\_RTC\_ALMB\_TIME\_FORMAT\_AM AM or 24-hour format

LL\_RTC\_ALMB\_TIME\_FORMAT\_PM PM

**RTC Alarm B Date WeekDay**

LL\_RTC\_ALMB\_DATEWEEKDAYSEL\_DATE Alarm B Date is selected

LL\_RTC\_ALMB\_DATEWEEKDAYSEL\_WEEKDAY Alarm B WeekDay is selected

**BACKUP**

LL\_RTC\_BKP\_DR0

LL\_RTC\_BKP\_DR1

LL\_RTC\_BKP\_DR2

LL\_RTC\_BKP\_DR3

LL\_RTC\_BKP\_DR4

LL\_RTC\_BKP\_DR5

LL\_RTC\_BKP\_DR6

LL\_RTC\_BKP\_DR7

LL\_RTC\_BKP\_DR8

LL\_RTC\_BKP\_DR9

LL\_RTC\_BKP\_DR10

LL\_RTC\_BKP\_DR11

LL\_RTC\_BKP\_DR12

LL\_RTC\_BKP\_DR13

LL\_RTC\_BKP\_DR14

LL\_RTC\_BKP\_DR15

**Calibration pulse insertion**

LL\_RTC\_CALIB\_INSERTPULSE\_NONE No RTCCLK pulses are added

LL\_RTC\_CALIB\_INSERTPULSE\_SET One RTCCLK pulse is effectively inserted every 2<sup>exp11</sup> pulses (frequency increased by 488.5 ppm)

**Calibration output**

LL\_RTC\_CALIB\_OUTPUT\_NONE Calibration output disabled

LL\_RTC\_CALIB\_OUTPUT\_1HZ Calibration output is 1 Hz

LL\_RTC\_CALIB\_OUTPUT\_512HZ Calibration output is 512 Hz

**Calibration period**

LL\_RTC\_CALIB\_PERIOD\_32SEC Use a 32-second calibration cycle period

LL\_RTC\_CALIB\_PERIOD\_16SEC Use a 16-second calibration cycle period

LL\_RTC\_CALIB\_PERIOD\_8SEC Use a 8-second calibration cycle period

**FORMAT**

LL\_RTC\_FORMAT\_BIN Binary data format

LL\_RTC\_FORMAT\_BCD BCD data format

**Get Flags Defines**

LL\_RTC\_ISR\_RECALPF

LL\_RTC\_ISR\_TAMP3F

LL\_RTC\_ISR\_TAMP2F

LL\_RTC\_ISR\_TAMP1F

LL\_RTC\_ISR\_TSOF

LL\_RTC\_ISR\_TSOF

LL\_RTC\_ISR\_WUTF

LL\_RTC\_ISR\_ALRBF

LL\_RTC\_ISR\_ALRAF

LL\_RTC\_ISR\_INITF

LL\_RTC\_ISR\_RSF

LL\_RTC\_ISR\_INITS

LL\_RTC\_ISR\_SHPF

LL\_RTC\_ISR\_WUTWF

LL\_RTC\_ISR\_ALRBWF

LL\_RTC\_ISR\_ALRAWF

**HOUR FORMAT**

LL\_RTC\_HOURFORMAT\_24HOUR 24 hour/day format

LL\_RTC\_HOURFORMAT\_AMPM AM/PM hour format

**IT Defines**

LL\_RTC\_CR\_TSIE

LL\_RTC\_CR\_WUTIE

LL\_RTC\_CR\_ALRBIE

LL\_RTC\_CR\_ALRAIE

LL\_RTC\_TAFCR\_TAMPIE

**MONTH**

LL\_RTC\_MONTH\_JANUARY January

LL\_RTC\_MONTH\_FEBRUARY February

LL\_RTC\_MONTH\_MARCH March

LL\_RTC\_MONTH\_APRIIL April

LL\_RTC\_MONTH\_MAY May

LL\_RTC\_MONTH\_JUNE June

LL\_RTC\_MONTH\_JULY July

LL\_RTC\_MONTH\_AUGUST August

LL\_RTC\_MONTH\_SEPTMBER September

LL\_RTC\_MONTH\_OCTOBER October

LL\_RTC\_MONTH\_NOVEMBER November

LL\_RTC\_MONTH\_DECEMBER December

**OUTPUT POLARITY PIN**

LL\_RTC\_OUTPUTPOLARITY\_PIN\_HIGH Pin is high when ALRAF/ALRBF/WUTF is asserted (depending on OSEL)

LL\_RTC\_OUTPUTPOLARITY\_PIN\_LOW Pin is low when ALRAF/ALRBF/WUTF is asserted (depending on OSEL)

**PIN**

LL\_RTC\_PIN\_PC13 PC13 is forced to push-pull output if all RTC alternate functions are disabled

LL\_RTC\_PIN\_PC14 PC14 is forced to push-pull output if LSE is disabled

LL\_RTC\_PIN\_PC15 PC15 is forced to push-pull output if LSE is disabled

**SHIFT SECOND**

LL\_RTC\_SHIFT\_SECOND\_DELAY

LL\_RTC\_SHIFT\_SECOND\_ADVANCE

**TAMPER**

LL\_RTC\_TAMPER\_1 RTC\_TAMP1 input detection

LL\_RTC\_TAMPER\_2 RTC\_TAMP2 input detection

LL\_RTC\_TAMPER\_3 RTC\_TAMP3 input detection

**TAMPER ACTIVE LEVEL**

|                                 |                                                                                                                     |
|---------------------------------|---------------------------------------------------------------------------------------------------------------------|
| LL_RTC_TAMPER_ACTIVELEVEL_TAMP1 | RTC_TAMP1 input falling edge (if TAMPFLT = 00) or staying high (if TAMPFLT != 00) triggers a tamper detection event |
| LL_RTC_TAMPER_ACTIVELEVEL_TAMP2 | RTC_TAMP2 input falling edge (if TAMPFLT = 00) or staying high (if TAMPFLT != 00) triggers a tamper detection event |
| LL_RTC_TAMPER_ACTIVELEVEL_TAMP3 | RTC_TAMP3 input falling edge (if TAMPFLT = 00) or staying high (if TAMPFLT != 00) triggers a tamper detection event |

**TAMPER DURATION**

|                                |                                                                    |
|--------------------------------|--------------------------------------------------------------------|
| LL_RTC_TAMPER_DURATION_1RTCCLK | Tamper pins are pre-charged before sampling during 1 RTCCLK cycle  |
| LL_RTC_TAMPER_DURATION_2RTCCLK | Tamper pins are pre-charged before sampling during 2 RTCCLK cycles |
| LL_RTC_TAMPER_DURATION_4RTCCLK | Tamper pins are pre-charged before sampling during 4 RTCCLK cycles |
| LL_RTC_TAMPER_DURATION_8RTCCLK | Tamper pins are pre-charged before sampling during 8 RTCCLK cycles |

**TAMPER FILTER**

|                              |                                                                      |
|------------------------------|----------------------------------------------------------------------|
| LL_RTC_TAMPER_FILTER_DISABLE | Tamper filter is disabled                                            |
| LL_RTC_TAMPER_FILTER_2SAMPLE | Tamper is activated after 2 consecutive samples at the active level  |
| LL_RTC_TAMPER_FILTER_4SAMPLE | Tamper is activated after 4 consecutive samples at the active level  |
| LL_RTC_TAMPER_FILTER_8SAMPLE | Tamper is activated after 8 consecutive samples at the active level. |

**TAMPER MASK**

|                            |                                                                                                                                     |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| LL_RTC_TAMPER_MASK_TAMPER1 | Tamper 1 event generates a trigger event. TAMP1F is masked and internally cleared by hardware. The backup registers are not erased  |
| LL_RTC_TAMPER_MASK_TAMPER2 | Tamper 2 event generates a trigger event. TAMP2F is masked and internally cleared by hardware. The backup registers are not erased. |
| LL_RTC_TAMPER_MASK_TAMPER3 | Tamper 3 event generates a trigger event. TAMP3F is masked and internally cleared by hardware. The backup registers are not erased  |

**TAMPER NO ERASE**

|                               |                                                     |
|-------------------------------|-----------------------------------------------------|
| LL_RTC_TAMPER_NOERASE_TAMPER1 | Tamper 1 event does not erase the backup registers. |
| LL_RTC_TAMPER_NOERASE_TAMPER2 | Tamper 2 event does not erase the backup registers. |

**LL\_RTC\_TAMPER\_NOERASE\_TAMPER3** Tamper 3 event does not erase the backup registers.

#### **TAMPER SAMPLING FREQUENCY DIVIDER**

|                                  |                                                                         |
|----------------------------------|-------------------------------------------------------------------------|
| LL_RTC_TAMPER_SAMPLFREQDIV_32768 | Each of the tamper inputs are sampled with a frequency = RTCCLK / 32768 |
| LL_RTC_TAMPER_SAMPLFREQDIV_16384 | Each of the tamper inputs are sampled with a frequency = RTCCLK / 16384 |
| LL_RTC_TAMPER_SAMPLFREQDIV_8192  | Each of the tamper inputs are sampled with a frequency = RTCCLK / 8192  |
| LL_RTC_TAMPER_SAMPLFREQDIV_4096  | Each of the tamper inputs are sampled with a frequency = RTCCLK / 4096  |
| LL_RTC_TAMPER_SAMPLFREQDIV_2048  | Each of the tamper inputs are sampled with a frequency = RTCCLK / 2048  |
| LL_RTC_TAMPER_SAMPLFREQDIV_1024  | Each of the tamper inputs are sampled with a frequency = RTCCLK / 1024  |
| LL_RTC_TAMPER_SAMPLFREQDIV_512   | Each of the tamper inputs are sampled with a frequency = RTCCLK / 512   |
| LL_RTC_TAMPER_SAMPLFREQDIV_256   | Each of the tamper inputs are sampled with a frequency = RTCCLK / 256   |

#### **TIMESTAMP EDGE**

|                               |                                                      |
|-------------------------------|------------------------------------------------------|
| LL_RTC_TIMESTAMP_EDGE_RISING  | RTC_TS input rising edge generates a timestamp event |
| LL_RTC_TIMESTAMP_EDGE_FALLING | RTC_TS input falling edge generates a timestamp even |

#### **TIME FORMAT**

|                             |                      |
|-----------------------------|----------------------|
| LL_RTC_TIME_FORMAT_AM_OR_24 | AM or 24-hour format |
| LL_RTC_TIME_FORMAT_PM       | PM                   |

#### **TIMESTAMP TIME FORMAT**

|                          |                      |
|--------------------------|----------------------|
| LL_RTC_TS_TIME_FORMAT_AM | AM or 24-hour format |
| LL_RTC_TS_TIME_FORMAT_PM | PM                   |

#### **WAKEUP CLOCK DIV**

|                               |                                                                                       |
|-------------------------------|---------------------------------------------------------------------------------------|
| LL_RTC_WAKEUPCLOCK_DIV_16     | RTC/16 clock is selected                                                              |
| LL_RTC_WAKEUPCLOCK_DIV_8      | RTC/8 clock is selected                                                               |
| LL_RTC_WAKEUPCLOCK_DIV_4      | RTC/4 clock is selected                                                               |
| LL_RTC_WAKEUPCLOCK_DIV_2      | RTC/2 clock is selected                                                               |
| LL_RTC_WAKEUPCLOCK_CKSPRE     | ck_spre (usually 1 Hz) clock is selected                                              |
| LL_RTC_WAKEUPCLOCK_CKSPRE_WUT | ck_spre (usually 1 Hz) clock is selected and 2exp16 is added to the WUT counter value |

#### **WEEK DAY**

|                       |        |
|-----------------------|--------|
| LL_RTC_WEEKDAY_MONDAY | Monday |
|-----------------------|--------|

---

|                          |           |
|--------------------------|-----------|
| LL_RTC_WEEKDAY_TUESDAY   | Tuesday   |
| LL_RTC_WEEKDAY_WEDNESDAY | Wednesday |
| LL_RTC_WEEKDAY_THURSDAY  | Thrusday  |
| LL_RTC_WEEKDAY_FRIDAY    | Friday    |
| LL_RTC_WEEKDAY_SATURDAY  | Saturday  |
| LL_RTC_WEEKDAY_SUNDAY    | Sunday    |

**Convert helper Macros****\_\_LL\_RTC\_CONVERT\_BIN2BCD** **Description:**

- Helper macro to convert a value from 2 digit decimal format to BCD format.

**Parameters:**

- \_\_VALUE\_\_: Byte to be converted

**Return value:**

- Converted: byte

**\_\_LL\_RTC\_CONVERT\_BCD2BIN** **Description:**

- Helper macro to convert a value from BCD format to 2 digit decimal format.

**Parameters:**

- \_\_VALUE\_\_: BCD value to be converted

**Return value:**

- Converted: byte

**Date helper Macros****\_\_LL\_RTC\_GET\_WEEKDAY** **Description:**

- Helper macro to retrieve weekday.

**Parameters:**

- \_\_RTC\_DATE\_\_: Date returned by

**Return value:**

- Returned: value can be one of the following values:
  - LL\_RTC\_WEEKDAY\_MONDAY
  - LL\_RTC\_WEEKDAY\_TUESDAY
  - LL\_RTC\_WEEKDAY\_WEDNESDAY
  - LL\_RTC\_WEEKDAY\_THURSDAY
  - LL\_RTC\_WEEKDAY\_FRIDAY
  - LL\_RTC\_WEEKDAY\_SATURDAY
  - LL\_RTC\_WEEKDAY\_SUNDAY

**\_\_LL\_RTC\_GET\_YEAR****Description:**

- Helper macro to retrieve Year in BCD format.

**Parameters:**

- \_\_RTC\_DATE\_\_: Value returned by

---

**\_LL\_RTC\_GET\_MONTH****Return value:**

- Year: in BCD format (0x00 . . . 0x99)

**Description:**

- Helper macro to retrieve Month in BCD format.

**Parameters:**

- \_\_RTC\_DATE\_\_: Value returned by

**Return value:**

- Returned: value can be one of the following values:

- LL\_RTC\_MONTH\_JANUARY
- LL\_RTC\_MONTH\_FEBRUARY
- LL\_RTC\_MONTH\_MARCH
- LL\_RTC\_MONTH\_APRI
- LL\_RTC\_MONTH\_MAY
- LL\_RTC\_MONTH\_JUNE
- LL\_RTC\_MONTH\_JULY
- LL\_RTC\_MONTH\_AUGUST
- LL\_RTC\_MONTH\_SEPTEMBER
- LL\_RTC\_MONTH\_OCTOBER
- LL\_RTC\_MONTH\_NOVEMBER
- LL\_RTC\_MONTH\_DECEMBER

**\_LL\_RTC\_GET\_DAY****Description:**

- Helper macro to retrieve Day in BCD format.

**Parameters:**

- \_\_RTC\_DATE\_\_: Value returned by

**Return value:**

- Day: in BCD format (0x01 . . . 0x31)

**Time helper Macros****\_LL\_RTC\_GET\_HOUR****Description:**

- Helper macro to retrieve hour in BCD format.

**Parameters:**

- \_\_RTC\_TIME\_\_: RTC time returned by

**Return value:**

- Hours: in BCD format (0x01 . . . 0x12 or between Min\_Data=0x00 and Max\_Data=0x23)

**\_LL\_RTC\_GET\_MINUTE****Description:**

- Helper macro to retrieve minute in BCD format.

**Parameters:**

- \_\_RTC\_TIME\_\_: RTC time returned by

**Return value:**

- Minutes: in BCD format (0x00 . . . 0x59)

**\_LL\_RTC\_GET\_SECOND    Description:**

- Helper macro to retrieve second in BCD format.

**Parameters:**

- \_\_RTC\_TIME\_\_: RTC time returned by

**Return value:**

- Seconds: in format (0x00...0x59)

**Common Write and read registers Macros****LL\_RTC\_WriteReg    Description:**

- Write a value in RTC register.

**Parameters:**

- \_\_INSTANCE\_\_: RTC Instance
- \_\_REG\_\_: Register to be written
- \_\_VALUE\_\_: Value to be written in the register

**Return value:**

- None

**LL\_RTC\_ReadReg    Description:**

- Read a value in RTC register.

**Parameters:**

- \_\_INSTANCE\_\_: RTC Instance
- \_\_REG\_\_: Register to be read

**Return value:**

- Register: value

## 76 LL SPI Generic Driver

### 76.1 SPI Firmware driver registers structures

#### 76.1.1 LL\_SPI\_InitTypeDef

##### Data Fields

- *uint32\_t TransferDirection*
- *uint32\_t Mode*
- *uint32\_t DataWidth*
- *uint32\_t ClockPolarity*
- *uint32\_t ClockPhase*
- *uint32\_t NSS*
- *uint32\_t BaudRate*
- *uint32\_t BitOrder*
- *uint32\_t CRCCalculation*
- *uint32\_t CRCPoly*

##### Field Documentation

- ***uint32\_t LL\_SPI\_InitTypeDef::TransferDirection***  
Specifies the SPI unidirectional or bidirectional data mode. This parameter can be a value of [\*\*SPI\\_LL\\_EC\\_TRANSFER\\_MODE\*\*](#). This feature can be modified afterwards using unitary function [\*\*LL\\_SPI\\_SetTransferDirection\(\)\*\*](#).
  - ***uint32\_t LL\_SPI\_InitTypeDef::Mode***  
Specifies the SPI mode (Master/Slave). This parameter can be a value of [\*\*SPI\\_LL\\_EC\\_MODE\*\*](#). This feature can be modified afterwards using unitary function [\*\*LL\\_SPI\\_SetMode\(\)\*\*](#).
  - ***uint32\_t LL\_SPI\_InitTypeDef::DataWidth***  
Specifies the SPI data width. This parameter can be a value of [\*\*SPI\\_LL\\_EC\\_DATAWIDTH\*\*](#). This feature can be modified afterwards using unitary function [\*\*LL\\_SPI\\_SetDataWidth\(\)\*\*](#).
  - ***uint32\_t LL\_SPI\_InitTypeDef::ClockPolarity***  
Specifies the serial clock steady state. This parameter can be a value of [\*\*SPI\\_LL\\_EC\\_POLARITY\*\*](#). This feature can be modified afterwards using unitary function [\*\*LL\\_SPI\\_SetClockPolarity\(\)\*\*](#).
  - ***uint32\_t LL\_SPI\_InitTypeDef::ClockPhase***  
Specifies the clock active edge for the bit capture. This parameter can be a value of [\*\*SPI\\_LL\\_EC\\_PHASE\*\*](#). This feature can be modified afterwards using unitary function [\*\*LL\\_SPI\\_SetClockPhase\(\)\*\*](#).
  - ***uint32\_t LL\_SPI\_InitTypeDef::NSS***  
Specifies whether the NSS signal is managed by hardware (NSS pin) or by software using the SSI bit. This parameter can be a value of [\*\*SPI\\_LL\\_EC\\_NSS\\_MODE\*\*](#). This feature can be modified afterwards using unitary function [\*\*LL\\_SPI\\_SetNSSMode\(\)\*\*](#).
  - ***uint32\_t LL\_SPI\_InitTypeDef::BaudRate***  
Specifies the BaudRate prescaler value which will be used to configure the transmit and receive SCK clock. This parameter can be a value of [\*\*SPI\\_LL\\_EC\\_BAUDRATEPRESCALER\*\*](#).
- Note:** The communication clock is derived from the master clock. The slave clock does not need to be set. This feature can be modified afterwards using unitary function [\*\*LL\\_SPI\\_SetBaudRatePrescaler\(\)\*\*](#).

- **`uint32_t LL_SPI_InitTypeDef::BitOrder`**  
Specifies whether data transfers start from MSB or LSB bit. This parameter can be a value of `SPI_LL_EC_BIT_ORDER`. This feature can be modified afterwards using unitary function `LL_SPI_SetTransferBitOrder()`.
- **`uint32_t LL_SPI_InitTypeDef::CRCCalculation`**  
Specifies if the CRC calculation is enabled or not. This parameter can be a value of `SPI_LL_EC_CRC_CALCULATION`. This feature can be modified afterwards using unitary functions `LL_SPI_EnableCRC()` and `LL_SPI_DisableCRC()`.
- **`uint32_t LL_SPI_InitTypeDef::CRCPoly`**  
Specifies the polynomial used for the CRC calculation. This parameter must be a number between Min\_Data = 0x00 and Max\_Data = 0xFFFF. This feature can be modified afterwards using unitary function `LL_SPI_SetCRCPolynomial()`.

## 76.2 SPI Firmware driver API description

### 76.2.1 Detailed description of functions

#### LL\_SPI\_Enable

|                                                   |                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_Enable (SPI_TypeDef * SPIx)</code>          |
| Function description                              | Enable SPI peripheral.                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 SPE LL_SPI_Enable</li> </ul>     |

#### LL\_SPI\_Disable

|                                                   |                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_Disable (SPI_TypeDef * SPIx)</code>                                                               |
| Function description                              | Disable SPI peripheral.                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• When disabling the SPI, follow the procedure described in the Reference Manual.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 SPE LL_SPI_Disable</li> </ul>                                                          |

#### LL\_SPI\_IsEnabled

|                                     |                                                                                    |
|-------------------------------------|------------------------------------------------------------------------------------|
| Function name                       | <code>__STATIC_INLINE uint32_t LL_SPI_IsEnabled (SPI_TypeDef * SPIx)</code>        |
| Function description                | Check if SPI peripheral is enabled.                                                |
| Parameters                          | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>      |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• CR1 SPE LL_SPI_IsEnabled</li> </ul>       |

reference:

### **LL\_SPI\_SetMode**

|                                                   |                                                                                                                                                                                                                                                               |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_SetMode (SPI_TypeDef * SPIx,<br/>uint32_t Mode)</code>                                                                                                                                                                      |
| Function description                              | Set SPI operation mode to Master or Slave.                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>Mode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_MODE_MASTER</li> <li>– LL_SPI_MODE_SLAVE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit should not be changed when communication is ongoing.</li> </ul>                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 MSTR LL_SPI_SetMode</li> <li>• CR1 SSI LL_SPI_SetMode</li> </ul>                                                                                                                                                 |

### **LL\_SPI\_GetMode**

|                                                   |                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_GetMode (SPI_TypeDef * SPIx)</code>                                                                                                                                           |
| Function description                              | Get SPI operation mode (Master or Slave)                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_MODE_MASTER</li> <li>– LL_SPI_MODE_SLAVE</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 MSTR LL_SPI_GetMode</li> <li>• CR1 SSI LL_SPI_GetMode</li> </ul>                                                                                                       |

### **LL\_SPI\_SetStandard**

|                                     |                                                                                                                                                                                                                                                                          |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                       | <code>__STATIC_INLINE void LL_SPI_SetStandard (SPI_TypeDef * SPIx, uint32_t Standard)</code>                                                                                                                                                                             |
| Function description                | Set serial protocol used.                                                                                                                                                                                                                                                |
| Parameters                          | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>Standard:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_PROTOCOL_MOTOROLA</li> <li>– LL_SPI_PROTOCOL_TI</li> </ul> </li> </ul> |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                          |
| Notes                               | <ul style="list-style-type: none"> <li>• This bit should be written only when SPI is disabled (SPE = 0) for correct operation.</li> </ul>                                                                                                                                |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• CR2 FRF LL_SPI_SetStandard</li> </ul>                                                                                                                                                                                           |

reference:

### **LL\_SPI\_GetStandard**

|                                                   |                                                                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_GetStandard (SPI_TypeDef * SPIx)</code>                                                                                                                                                        |
| Function description                              | Get serial protocol used.                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_SPI_PROTOCOL_MOTOROLA</li> <li>- LL_SPI_PROTOCOL_TI</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 FRF LL_SPI_GetStandard</li> </ul>                                                                                                                                                       |

### **LL\_SPI\_SetClockPhase**

|                                                   |                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_SetClockPhase (SPI_TypeDef * SPIx, uint32_t ClockPhase)</code>                                                                                                                                                                               |
| Function description                              | Set clock phase.                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>ClockPhase:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_SPI_PHASE_1EDGE</li> <li>- LL_SPI_PHASE_2EDGE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit should not be changed when communication is ongoing. This bit is not used in SPI TI mode.</li> </ul>                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 CPHA LL_SPI_SetClockPhase</li> </ul>                                                                                                                                                                                              |

### **LL\_SPI\_GetClockPhase**

|                                                   |                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_GetClockPhase (SPI_TypeDef * SPIx)</code>                                                                                                                                                |
| Function description                              | Get clock phase.                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_SPI_PHASE_1EDGE</li> <li>- LL_SPI_PHASE_2EDGE</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 CPHA LL_SPI_GetClockPhase</li> </ul>                                                                                                                                              |

**LL\_SPI\_SetClockPolarity**

|                                                   |                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_SPI_SetClockPolarity(SPI_TypeDef * SPIx, uint32_t ClockPolarity)</code>                                                                                                                                                                       |
| Function description                              | Set clock polarity.                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>ClockPolarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_POLARITY_LOW</li> <li>– LL_SPI_POLARITY_HIGH</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit should not be changed when communication is ongoing. This bit is not used in SPI TI mode.</li> </ul>                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 CPOL LL_SPI_SetClockPolarity</li> </ul>                                                                                                                                                                                       |

**LL\_SPI\_GetClockPolarity**

|                                                   |                                                                                                                                                                                                                         |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_SPI_GetClockPolarity(SPI_TypeDef * SPIx)</code>                                                                                                                                        |
| Function description                              | Get clock polarity.                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_POLARITY_LOW</li> <li>– LL_SPI_POLARITY_HIGH</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 CPOL LL_SPI_GetClockPolarity</li> </ul>                                                                                                                                    |

**LL\_SPI\_SetBaudRatePrescaler**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_SPI_SetBaudRatePrescaler(SPI_TypeDef * SPIx, uint32_t BaudRate)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | Set baud rate prescaler.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>BaudRate:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_BAUDRATEPRESCALER_DIV2</li> <li>– LL_SPI_BAUDRATEPRESCALER_DIV4</li> <li>– LL_SPI_BAUDRATEPRESCALER_DIV8</li> <li>– LL_SPI_BAUDRATEPRESCALER_DIV16</li> <li>– LL_SPI_BAUDRATEPRESCALER_DIV32</li> <li>– LL_SPI_BAUDRATEPRESCALER_DIV64</li> <li>– LL_SPI_BAUDRATEPRESCALER_DIV128</li> <li>– LL_SPI_BAUDRATEPRESCALER_DIV256</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

---

|                                             |                                                                                                                                                   |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>These bits should not be changed when communication is ongoing. SPI BaudRate = fPCLK/Prescaler.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 BR LL_SPI_SetBaudRatePrescaler</li> </ul>                                                              |

### LL\_SPI\_GetBaudRatePrescaler

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE uint32_t LL_SPI_GetBaudRatePrescaler(SPI_TypeDef * SPIx)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description                        | Get baud rate prescaler.                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Return values                               | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>LL_SPI_BAUDRATEPRESCALER_DIV2</li> <li>LL_SPI_BAUDRATEPRESCALER_DIV4</li> <li>LL_SPI_BAUDRATEPRESCALER_DIV8</li> <li>LL_SPI_BAUDRATEPRESCALER_DIV16</li> <li>LL_SPI_BAUDRATEPRESCALER_DIV32</li> <li>LL_SPI_BAUDRATEPRESCALER_DIV64</li> <li>LL_SPI_BAUDRATEPRESCALER_DIV128</li> <li>LL_SPI_BAUDRATEPRESCALER_DIV256</li> </ul> </li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 BR LL_SPI_GetBaudRatePrescaler</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                            |

### LL\_SPI\_SetTransferBitOrder

|                                             |                                                                                                                                                                                                                                                                  |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE void LL_SPI_SetTransferBitOrder(SPI_TypeDef * SPIx, uint32_t BitOrder)</code></b>                                                                                                                                                       |
| Function description                        | Set transfer bit order.                                                                                                                                                                                                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> <li><b>BitOrder:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>LL_SPI_LSB_FIRST</li> <li>LL_SPI_MSB_FIRST</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                    |
| Notes                                       | <ul style="list-style-type: none"> <li>This bit should not be changed when communication is ongoing. This bit is not used in SPI TI mode.</li> </ul>                                                                                                             |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 LSBFIRST LL_SPI_SetTransferBitOrder</li> </ul>                                                                                                                                                                        |

### LL\_SPI\_GetTransferBitOrder

|                      |                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_SPI_GetTransferBitOrder(SPI_TypeDef * SPIx)</code></b> |
| Function description | Get transfer bit order.                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>                 |

|                                                   |                                                                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_SPI_LSB_FIRST</li> <li>– LL_SPI_MSB_FIRST</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 LSBFIRST LL_SPI_SetTransferDirection</li> </ul>                                                                                                                             |

### LL\_SPI\_SetTransferDirection

|                                                   |                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_SetTransferDirection(SPI_TypeDef * SPIx, uint32_t TransferDirection)</code>                                                                                                                                                                                                                                          |
| Function description                              | Set transfer direction mode.                                                                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>TransferDirection:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_SPI_FULL_DUPLEX</li> <li>– LL_SPI_SIMPLEX_RX</li> <li>– LL_SPI_HALF_DUPLEX_RX</li> <li>– LL_SPI_HALF_DUPLEX_TX</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• For Half-Duplex mode, Rx Direction is set by default. In master mode, the MOSI pin is used and in slave mode, the MISO pin is used for Half-Duplex.</li> </ul>                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 RXONLY LL_SPI_SetTransferDirection</li> <li>• CR1 BIDIMODE LL_SPI_SetTransferDirection</li> <li>• CR1 BIDIOE LL_SPI_SetTransferDirection</li> </ul>                                                                                                                                                       |

### LL\_SPI\_GetTransferDirection

|                                                   |                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_GetTransferDirection(SPI_TypeDef * SPIx)</code>                                                                                                                                                                                                           |
| Function description                              | Get transfer direction mode.                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_SPI_FULL_DUPLEX</li> <li>– LL_SPI_SIMPLEX_RX</li> <li>– LL_SPI_HALF_DUPLEX_RX</li> <li>– LL_SPI_HALF_DUPLEX_TX</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 RXONLY LL_SPI_SetTransferDirection</li> <li>• CR1 BIDIMODE LL_SPI_SetTransferDirection</li> <li>• CR1 BIDIOE LL_SPI_SetTransferDirection</li> </ul>                                                                                                |

### LL\_SPI\_SetDataWidth

|                      |                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_SPI_SetDataWidth(SPI_TypeDef * SPIx, uint32_t DataWidth)</code> |
| Function description | Set frame data width.                                                                         |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>DataWidth:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_DATAWIDTH_4BIT</li> <li>– LL_SPI_DATAWIDTH_5BIT</li> <li>– LL_SPI_DATAWIDTH_6BIT</li> <li>– LL_SPI_DATAWIDTH_7BIT</li> <li>– LL_SPI_DATAWIDTH_8BIT</li> <li>– LL_SPI_DATAWIDTH_9BIT</li> <li>– LL_SPI_DATAWIDTH_10BIT</li> <li>– LL_SPI_DATAWIDTH_11BIT</li> <li>– LL_SPI_DATAWIDTH_12BIT</li> <li>– LL_SPI_DATAWIDTH_13BIT</li> <li>– LL_SPI_DATAWIDTH_14BIT</li> <li>– LL_SPI_DATAWIDTH_15BIT</li> <li>– LL_SPI_DATAWIDTH_16BIT</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 DS LL_SPI_SetDataWidth</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

### LL\_SPI\_GetDataWidth

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_GetDataWidth(<br/>(SPI_TypeDef * SPIx)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Get frame data width.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_DATAWIDTH_4BIT</li> <li>– LL_SPI_DATAWIDTH_5BIT</li> <li>– LL_SPI_DATAWIDTH_6BIT</li> <li>– LL_SPI_DATAWIDTH_7BIT</li> <li>– LL_SPI_DATAWIDTH_8BIT</li> <li>– LL_SPI_DATAWIDTH_9BIT</li> <li>– LL_SPI_DATAWIDTH_10BIT</li> <li>– LL_SPI_DATAWIDTH_11BIT</li> <li>– LL_SPI_DATAWIDTH_12BIT</li> <li>– LL_SPI_DATAWIDTH_13BIT</li> <li>– LL_SPI_DATAWIDTH_14BIT</li> <li>– LL_SPI_DATAWIDTH_15BIT</li> <li>– LL_SPI_DATAWIDTH_16BIT</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 DS LL_SPI_SetDataWidth</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

**LL\_SPI\_SetRxFIFOThreshold**

|                                                   |                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_SetRxFIFOThreshold (SPI_TypeDef * SPIx, uint32_t Threshold)</code>                                                                                                                                                                           |
| Function description                              | Set threshold of RXFIFO that triggers an RXNE event.                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>Threshold:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_RX_FIFO_TH_HALF</li> <li>– LL_SPI_RX_FIFO_TH_QUARTER</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 FRXTH LL_SPI_SetRxFIFOThreshold</li> </ul>                                                                                                                                                                                        |

**LL\_SPI\_GetRxFIFOThreshold**

|                                                   |                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_GetRxFIFOThreshold (SPI_TypeDef * SPIx)</code>                                                                                                                                            |
| Function description                              | Get threshold of RXFIFO that triggers an RXNE event.                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_RX_FIFO_TH_HALF</li> <li>– LL_SPI_RX_FIFO_TH_QUARTER</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 FRXTH LL_SPI_GetRxFIFOThreshold</li> </ul>                                                                                                                                         |

**LL\_SPI\_EnableCRC**

|                                                   |                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_EnableCRC (SPI_TypeDef * SPIx)</code>                                                                   |
| Function description                              | Enable CRC.                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit should be written only when SPI is disabled (SPE = 0) for correct operation.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 CRCEN LL_SPI_EnableCRC</li> </ul>                                                            |

**LL\_SPI\_DisableCRC**

|                      |                                                                          |
|----------------------|--------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_SPI_DisableCRC (SPI_TypeDef * SPIx)</code> |
| Function description | Disable CRC.                                                             |

---

|                                                   |                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>This bit should be written only when SPI is disabled (SPE = 0) for correct operation.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 CRCEN LL_SPI_DisableCRC</li> </ul>                                                           |

### LL\_SPI\_IsEnabledCRC

|                                                   |                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_IsEnabledCRC (SPI_TypeDef * SPIx)</code>                                                          |
| Function description                              | Check if CRC is enabled.                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>This bit should be written only when SPI is disabled (SPE = 0) for correct operation.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 CRCEN LL_SPI_IsEnabledCRC</li> </ul>                                                         |

### LL\_SPI\_SetCRCWidth

|                                                   |                                                                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_SetCRCWidth (SPI_TypeDef * SPIx, uint32_t CRCLength)</code>                                                                                                                                                              |
| Function description                              | Set CRC Length.                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> <li><b>CRCLength:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_CRC_8BIT</li> <li>– LL_SPI_CRC_16BIT</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>This bit should be written only when SPI is disabled (SPE = 0) for correct operation.</li> </ul>                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 CRCL LL_SPI_SetCRCWidth</li> </ul>                                                                                                                                                                              |

### LL\_SPI\_GetCRCWidth

|                      |                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_SPI_GetCRCWidth (SPI_TypeDef * SPIx)</code>                                                                                                     |
| Function description | Get CRC Length.                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_SPI_CRC_8BIT</li> </ul> </li> </ul> |

- LL\_SPI\_CRC\_16BIT

- Reference Manual to  
LL API cross  
reference:
- CR1 CRCL LL\_SPI\_GetCRCWidth

### LL\_SPI\_SetCRCNext

|                                                   |                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_SetCRCNext (SPI_TypeDef * SPIx)</code>                                                                  |
| Function description                              | Set CRCNext to transfer CRC on the line.                                                                                                  |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>SPIx:</b> SPI Instance</li></ul>                                                               |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                             |
| Notes                                             | <ul style="list-style-type: none"><li>• This bit has to be written as soon as the last data is written in the SPIx_DR register.</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR1 CRCNEXT LL_SPI_SetCRCNext</li></ul>                                                           |

### LL\_SPI\_SetCRCPolynomial

|                                                   |                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_SetCRCPolynomial (SPI_TypeDef * SPIx, uint32_t CRCPoly)</code>                                                                                    |
| Function description                              | Set polynomial for CRC calculation.                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>SPIx:</b> SPI Instance</li><li>• <b>CRCPoly:</b> This parameter must be a number between Min_Data = 0x00 and Max_Data = 0xFFFF</li></ul> |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CRCPR CRCPOLY LL_SPI_SetCRCPolynomial</li></ul>                                                                                             |

### LL\_SPI\_GetCRCPolynomial

|                                                   |                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_GetCRCPolynomial (SPI_TypeDef * SPIx)</code>                                                 |
| Function description                              | Get polynomial for CRC calculation.                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>SPIx:</b> SPI Instance</li></ul>                                                        |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>Returned:</b> value is a number between Min_Data = 0x00 and Max_Data = 0xFFFF</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CRCPR CRCPOLY LL_SPI_GetCRCPolynomial</li></ul>                                            |

**LL\_SPI\_GetRxCRC**

|                                                   |                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_SPI_GetRxCRC (SPI_TypeDef * SPIx)</code>                                                            |
| Function description                              | Get Rx CRC.                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value is a number between Min_Data = 0x00 and Max_Data = 0xFFFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• RXCRCR RXCRC LL_SPI_GetRxCRC</li> </ul>                                                     |

**LL\_SPI\_GetTxCRC**

|                                                   |                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_SPI_GetTxCRC (SPI_TypeDef * SPIx)</code>                                                            |
| Function description                              | Get Tx CRC.                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value is a number between Min_Data = 0x00 and Max_Data = 0xFFFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TXCRCR TXCRC LL_SPI_GetTxCRC</li> </ul>                                                     |

**LL\_SPI\_SetNSSMode**

|                                                   |                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_SPI_SetNSSMode (SPI_TypeDef * SPIx, uint32_t NSS)</code>                                                                                                                                                                                                           |
| Function description                              | Set NSS mode.                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>NSS:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_SPI_NSS_SOFT</li> <li>- LL_SPI_NSS_HARD_INPUT</li> <li>- LL_SPI_NSS_HARD_OUTPUT</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>• LL_SPI_NSS_SOFT Mode is not used in SPI TI mode.</li> </ul>                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 SSM LL_SPI_SetNSSMode</li> <li>• CR2 SSOE LL_SPI_SetNSSMode</li> </ul>                                                                                                                                                                             |

**LL\_SPI\_GetNSSMode**

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_SPI_GetNSSMode (SPI_TypeDef * SPIx)</code>   |
| Function description | Get NSS mode.                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul> |

- |               |                                                                                                                                                                                                                                                                  |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_SPI_NSS_SOFT</li> <li>- LL_SPI_NSS_HARD_INPUT</li> <li>- LL_SPI_NSS_HARD_OUTPUT</li> </ul> </li> </ul> |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

- |                                                   |                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 SSM LL_SPI_GetNSSMode</li> <li>• CR2 SSOE LL_SPI_GetNSSMode</li> </ul> |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|

### **LL\_SPI\_EnableNSSPulseMgt**

- |                                                   |                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_SPI_EnableNSSPulseMgt(SPI_TypeDef * SPIx)</b>                                                                                |
| Function description                              | Enable NSS pulse management.                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit should not be changed when communication is ongoing. This bit is not used in SPI TI mode.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 NSSP LL_SPI_EnableNSSPulseMgt</li> </ul>                                                                  |

### **LL\_SPI\_DisableNSSPulseMgt**

- |                                                   |                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_SPI_DisableNSSPulseMgt(SPI_TypeDef * SPIx)</b>                                                                               |
| Function description                              | Disable NSS pulse management.                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit should not be changed when communication is ongoing. This bit is not used in SPI TI mode.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 NSSP LL_SPI_DisableNSSPulseMgt</li> </ul>                                                                 |

### **LL\_SPI\_IsEnabledNSSPulse**

- |                      |                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>_STATIC_INLINE uint32_t LL_SPI_IsEnabledNSSPulse(SPI_TypeDef * SPIx)</b>                                                                            |
| Function description | Check if NSS pulse is enabled.                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• This bit should not be changed when communication is ongoing. This bit is not used in SPI TI mode.</li> </ul> |

- Reference Manual to CR2 NSSP LL\_SPI\_IsEnabledNSSPulse  
LL API cross reference:

### **LL\_SPI\_IsActiveFlag\_RXNE**

Function name **\_STATIC\_INLINE uint32\_t LL\_SPI\_IsActiveFlag\_RXNE(SPI\_TypeDef \* SPIx)**

Function description Check if Rx buffer is not empty.

Parameters • **SPIx:** SPI Instance

Return values • **State:** of bit (1 or 0).

- Reference Manual to SR RXNE LL\_SPI\_IsActiveFlag\_RXNE  
LL API cross reference:

### **LL\_SPI\_IsActiveFlag\_TXE**

Function name **\_STATIC\_INLINE uint32\_t LL\_SPI\_IsActiveFlag\_TXE(SPI\_TypeDef \* SPIx)**

Function description Check if Tx buffer is empty.

Parameters • **SPIx:** SPI Instance

Return values • **State:** of bit (1 or 0).

- Reference Manual to SR TXE LL\_SPI\_IsActiveFlag\_TXE  
LL API cross reference:

### **LL\_SPI\_IsActiveFlag\_CRCERR**

Function name **\_STATIC\_INLINE uint32\_t LL\_SPI\_IsActiveFlag\_CRCERR(SPI\_TypeDef \* SPIx)**

Function description Get CRC error flag.

Parameters • **SPIx:** SPI Instance

Return values • **State:** of bit (1 or 0).

- Reference Manual to SR CRCERR LL\_SPI\_IsActiveFlag\_CRCERR  
LL API cross reference:

### **LL\_SPI\_IsActiveFlag\_MODF**

Function name **\_STATIC\_INLINE uint32\_t LL\_SPI\_IsActiveFlag\_MODF(SPI\_TypeDef \* SPIx)**

Function description Get mode fault error flag.

Parameters • **SPIx:** SPI Instance

Return values • **State:** of bit (1 or 0).

- Reference Manual to  
LL API cross  
reference:
- SR MODF LL\_SPI\_IsActiveFlag\_MODF

### LL\_SPI\_IsActiveFlag\_OVR

Function name `_STATIC_INLINE uint32_t LL_SPI_IsActiveFlag_OVR(SPI_TypeDef * SPIx)`

Function description Get overrun error flag.

Parameters • **SPIx:** SPI Instance

Return values • **State:** of bit (1 or 0).

- Reference Manual to  
LL API cross  
reference:
- SR OVR LL\_SPI\_IsActiveFlag\_OVR

### LL\_SPI\_IsActiveFlag\_BSY

Function name `_STATIC_INLINE uint32_t LL_SPI_IsActiveFlag_BSY(SPI_TypeDef * SPIx)`

Function description Get busy flag.

Parameters • **SPIx:** SPI Instance

Return values • **State:** of bit (1 or 0).

Notes • The BSY flag is cleared under any one of the following conditions:  
-When the SPI is correctly disabled  
-When a fault is detected in Master mode (MODF bit set to 1)  
-In Master mode, when it finishes a data transmission and no new data is ready to be sent  
-In Slave mode, when the BSY flag is set to '0' for at least one SPI clock cycle between each data transfer.

- Reference Manual to  
LL API cross  
reference:
- SR BSY LL\_SPI\_IsActiveFlag\_BSY

### LL\_SPI\_IsActiveFlag\_FRE

Function name `_STATIC_INLINE uint32_t LL_SPI_IsActiveFlag_FRE(SPI_TypeDef * SPIx)`

Function description Get frame format error flag.

Parameters • **SPIx:** SPI Instance

Return values • **State:** of bit (1 or 0).

- Reference Manual to  
LL API cross  
reference:
- SR FRE LL\_SPI\_IsActiveFlag\_FRE

**LL\_SPI\_GetRxFIFOLevel**

|                                                   |                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_GetRxFIFOLevel(SPI_TypeDef * SPIx)</code>                                                                                                                                                                                                                              |
| Function description                              | Get FIFO reception Level.                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_SPI_RX_FIFO_EMPTY</li> <li>- LL_SPI_RX_FIFO_QUARTER_FULL</li> <li>- LL_SPI_RX_FIFO_HALF_FULL</li> <li>- LL_SPI_RX_FIFO_FULL</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR FRLVL LL_SPI_GetRxFIFOLevel</li> </ul>                                                                                                                                                                                                                           |

**LL\_SPI\_GetTxFIFOLevel**

|                                                   |                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_GetTxFIFOLevel(SPI_TypeDef * SPIx)</code>                                                                                                                                                                                                                              |
| Function description                              | Get FIFO Transmission Level.                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_SPI_TX_FIFO_EMPTY</li> <li>- LL_SPI_TX_FIFO_QUARTER_FULL</li> <li>- LL_SPI_TX_FIFO_HALF_FULL</li> <li>- LL_SPI_TX_FIFO_FULL</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR FTLVL LL_SPI_GetTxFIFOLevel</li> </ul>                                                                                                                                                                                                                           |

**LL\_SPI\_ClearFlag\_CRCERR**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_ClearFlag_CRCERR(SPI_TypeDef * SPIx)</code>         |
| Function description                              | Clear CRC error flag.                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CRCERR LL_SPI_ClearFlag_CRCERR</li> </ul> |

**LL\_SPI\_ClearFlag\_MODF**

|                      |                                                                               |
|----------------------|-------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_SPI_ClearFlag_MODF(SPI_TypeDef * SPIx)</code>   |
| Function description | Clear mode fault error flag.                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul> |

|                                                   |                                                                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>Clearing this flag is done by a read access to the SPIx_SR register followed by a write access to the SPIx_CR1 register</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SR MODF LL_SPI_ClearFlag_MODF</li> </ul>                                                                                           |

### LL\_SPI\_ClearFlag\_OVR

|                                                   |                                                                                                                                                                         |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_SPI_ClearFlag_OVR (SPI_TypeDef * SPIx)</code>                                                                                              |
| Function description                              | Clear overrun error flag.                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>Clearing this flag is done by a read access to the SPIx_DR register followed by a read access to the SPIx_SR register</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SR OVR LL_SPI_ClearFlag_OVR</li> </ul>                                                                                           |

### LL\_SPI\_ClearFlag\_FRE

|                                                   |                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_SPI_ClearFlag_FRE (SPI_TypeDef * SPIx)</code>                               |
| Function description                              | Clear frame format error flag.                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>                              |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>Clearing this flag is done by reading SPIx_SR register</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SR FRE LL_SPI_ClearFlag_FRE</li> </ul>                            |

### LL\_SPI\_EnableIT\_ERR

|                                                   |                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_SPI_EnableIT_ERR (SPI_TypeDef * SPIx)</code>                                                                                                          |
| Function description                              | Enable error interrupt.                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>SPIx:</b> SPI Instance</li> </ul>                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>This bit controls the generation of an interrupt when an error condition occurs (CRCERR, OVR, MODF in SPI mode, FRE at TI mode).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR2 ERRIE LL_SPI_EnableIT_ERR</li> </ul>                                                                                                    |

**LL\_SPI\_EnableIT\_RXNE**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_EnableIT_RXNE (SPI_TypeDef * SPIx)</code>         |
| Function description                              | Enable Rx buffer not empty interrupt.                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RXNEIE LL_SPI_EnableIT_RXNE</li> </ul> |

**LL\_SPI\_EnableIT\_TXE**

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_EnableIT_TXE (SPI_TypeDef * SPIx)</code>        |
| Function description                              | Enable Tx buffer empty interrupt.                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 TXEIE LL_SPI_EnableIT_TXE</li> </ul> |

**LL\_SPI\_DisableIT\_ERR**

|                                                   |                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_DisableIT_ERR (SPI_TypeDef * SPIx)</code>                                                                                                          |
| Function description                              | Disable error interrupt.                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• This bit controls the generation of an interrupt when an error condition occurs (CRCERR, OVR, MODF in SPI mode, FRE at TI mode).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 ERRIE LL_SPI_DisableIT_ERR</li> </ul>                                                                                                   |

**LL\_SPI\_DisableIT\_RXNE**

|                                     |                                                                                      |
|-------------------------------------|--------------------------------------------------------------------------------------|
| Function name                       | <code>__STATIC_INLINE void LL_SPI_DisableIT_RXNE (SPI_TypeDef * SPIx)</code>         |
| Function description                | Disable Rx buffer not empty interrupt.                                               |
| Parameters                          | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>        |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• CR2 RXNEIE LL_SPI_DisableIT_RXNE</li> </ul> |

reference:

### LL\_SPI\_DisableIT\_TXE

|                                                   |                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_DisableIT_TXE (SPI_TypeDef * SPIx)</code>    |
| Function description                              | Disable Tx buffer empty interrupt.                                             |
| Parameters                                        | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>      |
| Return values                                     | <ul style="list-style-type: none"><li><b>None</b></li></ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR2 TXEIE LL_SPI_DisableIT_TXE</li></ul> |

### LL\_SPI\_IsEnabledIT\_ERR

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_IsEnabledIT_ERR (SPI_TypeDef * SPIx)</code> |
| Function description                              | Check if error interrupt is enabled.                                              |
| Parameters                                        | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>         |
| Return values                                     | <ul style="list-style-type: none"><li><b>State:</b> of bit (1 or 0).</li></ul>    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR2 ERRIE LL_SPI_IsEnabledIT_ERR</li></ul>  |

### LL\_SPI\_IsEnabledIT\_RXNE

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_IsEnabledIT_RXNE (SPI_TypeDef * SPIx)</code> |
| Function description                              | Check if Rx buffer not empty interrupt is enabled.                                 |
| Parameters                                        | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>          |
| Return values                                     | <ul style="list-style-type: none"><li><b>State:</b> of bit (1 or 0).</li></ul>     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR2 RXNEIE LL_SPI_IsEnabledIT_RXNE</li></ul> |

### LL\_SPI\_IsEnabledIT\_TXE

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_IsEnabledIT_TXE (SPI_TypeDef * SPIx)</code> |
| Function description                              | Check if Tx buffer empty interrupt.                                               |
| Parameters                                        | <ul style="list-style-type: none"><li><b>SPIx:</b> SPI Instance</li></ul>         |
| Return values                                     | <ul style="list-style-type: none"><li><b>State:</b> of bit (1 or 0).</li></ul>    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR2 TXEIE LL_SPI_IsEnabledIT_TXE</li></ul>  |

**LL\_SPI\_EnableDMAReq\_RX**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_SPI_EnableDMAReq_RX<br/>(SPI_TypeDef * SPIx)</code>       |
| Function description                              | Enable DMA Rx.                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RXDMAEN LL_SPI_EnableDMAReq_RX</li> </ul> |

**LL\_SPI\_DisableDMAReq\_RX**

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_SPI_DisableDMAReq_RX<br/>(SPI_TypeDef * SPIx)</code>       |
| Function description                              | Disable DMA Rx.                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RXDMAEN LL_SPI_DisableDMAReq_RX</li> </ul> |

**LL\_SPI\_IsEnabledDMAReq\_RX**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_SPI_IsEnabledDMAReq_RX<br/>(SPI_TypeDef * SPIx)</code>   |
| Function description                              | Check if DMA Rx is enabled.                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RXDMAEN LL_SPI_IsEnabledDMAReq_RX</li> </ul> |

**LL\_SPI\_EnableDMAReq\_TX**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_SPI_EnableDMAReq_TX<br/>(SPI_TypeDef * SPIx)</code>       |
| Function description                              | Enable DMA Tx.                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 TXDMAEN LL_SPI_EnableDMAReq_TX</li> </ul> |

**LL\_SPI\_DisableDMAReq\_TX**

Function name      **\_\_STATIC\_INLINE void LL\_SPI\_DisableDMAReq\_TX(SPI\_TypeDef \* SPIx)**

Function description      Disable DMA Tx.

Parameters      • **SPIx:** SPI Instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR2 TXDMAEN LL\_SPI\_DisableDMAReq\_TX

**LL\_SPI\_IsEnabledDMAReq\_TX**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_SPI\_IsEnabledDMAReq\_TX(SPI\_TypeDef \* SPIx)**

Function description      Check if DMA Tx is enabled.

Parameters      • **SPIx:** SPI Instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
• CR2 TXDMAEN LL\_SPI\_IsEnabledDMAReq\_TX

**LL\_SPI\_SetDMAParity\_RX**

Function name      **\_\_STATIC\_INLINE void LL\_SPI\_SetDMAParity\_RX(SPI\_TypeDef \* SPIx, uint32\_t Parity)**

Function description      Set parity of Last DMA reception.

Parameters      • **SPIx:** SPI Instance

- **Parity:** This parameter can be one of the following values:
  - LL\_SPI\_DMA\_PARITY\_ODD
  - LL\_SPI\_DMA\_PARITY EVEN

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR2 LDMARX LL\_SPI\_SetDMAParity\_RX

**LL\_SPI\_GetDMAParity\_RX**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_SPI\_GetDMAParity\_RX(SPI\_TypeDef \* SPIx)**

Function description      Get parity configuration for Last DMA reception.

Parameters      • **SPIx:** SPI Instance

Return values      • **Returned:** value can be one of the following values:
 

- LL\_SPI\_DMA\_PARITY\_ODD
- LL\_SPI\_DMA\_PARITY EVEN

- Reference Manual to  
LL API cross  
reference:
- CR2 LDMARX LL\_SPI\_SetDMAParity\_RX

### **LL\_SPI\_SetDMAParity\_TX**

|                                                   |                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SPI_SetDMAParity_TX(SPI_TypeDef * SPIx, uint32_t Parity)</code>                                                                                                                                                                           |
| Function description                              | Set parity of Last DMA transmission.                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>Parity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_SPI_DMA_PARITY_ODD</li> <li>- LL_SPI_DMA_PARITY_EVEN</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 LDMATX LL_SPI_SetDMAParity_TX</li> </ul>                                                                                                                                                                                   |

### **LL\_SPI\_GetDMAParity\_TX**

|                                                   |                                                                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_GetDMAParity_TX(SPI_TypeDef * SPIx)</code>                                                                                                                                            |
| Function description                              | Get parity configuration for Last DMA transmission.                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_SPI_DMA_PARITY_ODD</li> <li>- LL_SPI_DMA_PARITY_EVEN</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 LDMATX LL_SPI_GetDMAParity_TX</li> </ul>                                                                                                                                       |

### **LL\_SPI\_DMA\_GetRegAddr**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SPI_DMA_GetRegAddr(SPI_TypeDef * SPIx)</code>      |
| Function description                              | Get the data register address used for DMA transfer.                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Address:</b> of data register</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DR DR LL_SPI_DMA_GetRegAddr</li> </ul>      |

**LL\_SPI\_ReceiveData8**

|                                             |                                                                                                                |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint8_t LL_SPI_ReceiveData8 (SPI_TypeDef * SPIx)</code>                                  |
| Function description                        | Read 8-Bits in the data register.                                                                              |
| Parameters                                  | <ul style="list-style-type: none"><li>• <b>SPIx:</b> SPI Instance</li></ul>                                    |
| Return values                               | <ul style="list-style-type: none"><li>• <b>RxData:</b> Value between Min_Data=0x00 and Max_Data=0xFF</li></ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>• DR DR LL_SPI_ReceiveData8</li></ul>                                    |

**LL\_SPI\_ReceiveData16**

|                                             |                                                                                                                  |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint16_t LL_SPI_ReceiveData16 (SPI_TypeDef * SPIx)</code>                                  |
| Function description                        | Read 16-Bits in the data register.                                                                               |
| Parameters                                  | <ul style="list-style-type: none"><li>• <b>SPIx:</b> SPI Instance</li></ul>                                      |
| Return values                               | <ul style="list-style-type: none"><li>• <b>RxData:</b> Value between Min_Data=0x00 and Max_Data=0xFFFF</li></ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>• DR DR LL_SPI_ReceiveData16</li></ul>                                     |

**LL\_SPI\_TransmitData8**

|                                             |                                                                                                                                                    |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_SPI_TransmitData8 (SPI_TypeDef * SPIx, uint8_t TxData)</code>                                                        |
| Function description                        | Write 8-Bits in the data register.                                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"><li>• <b>SPIx:</b> SPI Instance</li><li>• <b>TxData:</b> Value between Min_Data=0x00 and Max_Data=0xFF</li></ul> |
| Return values                               | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"><li>• DR DR LL_SPI_TransmitData8</li></ul>                                                                       |

**LL\_SPI\_TransmitData16**

|                      |                                                                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_SPI_TransmitData16 (SPI_TypeDef * SPIx, uint16_t TxData)</code>                                                        |
| Function description | Write 16-Bits in the data register.                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"><li>• <b>SPIx:</b> SPI Instance</li><li>• <b>TxData:</b> Value between Min_Data=0x00 and Max_Data=0xFFFF</li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                        |

Reference Manual to  
LL API cross  
reference:

- DR DR LL\_SPI\_TransmitData16

### **LL\_SPI\_DeInit**

|                      |                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_SPI_DeInit (SPI_TypeDef * SPIx)</b>                                                                                                                                                                                                         |
| Function description | De-initialize the SPI registers to their default reset values.                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> </ul>                                                                                                                                                                                 |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: SPI registers are de-initialized</li> <li>– ERROR: SPI registers are not de-initialized</li> </ul> </li> </ul> |

### **LL\_SPI\_Init**

|                      |                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_SPI_Init (SPI_TypeDef * SPIx,<br/>LL_SPI_InitTypeDef * SPI_InitStruct)</b>                                                                                                                                                                                 |
| Function description | Initialize the SPI registers according to the specified parameters in SPI_InitStruct.                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPIx:</b> SPI Instance</li> <li>• <b>SPI_InitStruct:</b> pointer to a LL_SPI_InitTypeDef structure</li> </ul>                                                                                                                    |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value. (Return always SUCCESS)</li> </ul>                                                                                                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• As some bits in SPI configuration registers can only be written when the SPI is disabled (SPI_CR1_SPE bit =0), SPI IP should be in disabled state prior calling this function. Otherwise, ERROR result will be returned.</li> </ul> |

### **LL\_SPI\_StructInit**

|                      |                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_SPI_StructInit (LL_SPI_InitTypeDef * SPI_InitStruct)</b>                                                                                              |
| Function description | Set each LL_SPI_InitTypeDef field to default value.                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>SPI_InitStruct:</b> pointer to a LL_SPI_InitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                  |

## 76.3 SPI Firmware driver defines

### 76.3.1 SPI

#### **Baud Rate Prescaler**

|                                |                                    |
|--------------------------------|------------------------------------|
| LL_SPI_BAUDRATEPRESCALER_DIV2  | BaudRate control equal to fPCLK/2  |
| LL_SPI_BAUDRATEPRESCALER_DIV4  | BaudRate control equal to fPCLK/4  |
| LL_SPI_BAUDRATEPRESCALER_DIV8  | BaudRate control equal to fPCLK/8  |
| LL_SPI_BAUDRATEPRESCALER_DIV16 | BaudRate control equal to fPCLK/16 |

|                                 |                                     |
|---------------------------------|-------------------------------------|
| LL_SPI_BAUDRATEPRESCALER_DIV32  | BaudRate control equal to fPCLK/32  |
| LL_SPI_BAUDRATEPRESCALER_DIV64  | BaudRate control equal to fPCLK/64  |
| LL_SPI_BAUDRATEPRESCALER_DIV128 | BaudRate control equal to fPCLK/128 |
| LL_SPI_BAUDRATEPRESCALER_DIV256 | BaudRate control equal to fPCLK/256 |

**Transmission Bit Order**

|                  |                                                 |
|------------------|-------------------------------------------------|
| LL_SPI_LSB_FIRST | Data is transmitted/received with the LSB first |
| LL_SPI_MSB_FIRST | Data is transmitted/received with the MSB first |

**CRC Calculation**

|                               |                          |
|-------------------------------|--------------------------|
| LL_SPI_CRCCALCULATION_DISABLE | CRC calculation disabled |
| LL_SPI_CRCCALCULATION_ENABLE  | CRC calculation enabled  |

**CRC Length**

|                  |                   |
|------------------|-------------------|
| LL_SPI_CRC_8BIT  | 8-bit CRC length  |
| LL_SPI_CRC_16BIT | 16-bit CRC length |

**Datawidth**

|                        |                                       |
|------------------------|---------------------------------------|
| LL_SPI_DATAWIDTH_4BIT  | Data length for SPI transfer: 4 bits  |
| LL_SPI_DATAWIDTH_5BIT  | Data length for SPI transfer: 5 bits  |
| LL_SPI_DATAWIDTH_6BIT  | Data length for SPI transfer: 6 bits  |
| LL_SPI_DATAWIDTH_7BIT  | Data length for SPI transfer: 7 bits  |
| LL_SPI_DATAWIDTH_8BIT  | Data length for SPI transfer: 8 bits  |
| LL_SPI_DATAWIDTH_9BIT  | Data length for SPI transfer: 9 bits  |
| LL_SPI_DATAWIDTH_10BIT | Data length for SPI transfer: 10 bits |
| LL_SPI_DATAWIDTH_11BIT | Data length for SPI transfer: 11 bits |
| LL_SPI_DATAWIDTH_12BIT | Data length for SPI transfer: 12 bits |
| LL_SPI_DATAWIDTH_13BIT | Data length for SPI transfer: 13 bits |
| LL_SPI_DATAWIDTH_14BIT | Data length for SPI transfer: 14 bits |
| LL_SPI_DATAWIDTH_15BIT | Data length for SPI transfer: 15 bits |
| LL_SPI_DATAWIDTH_16BIT | Data length for SPI transfer: 16 bits |

**DMA Parity**

|                        |                        |
|------------------------|------------------------|
| LL_SPI_DMA_PARITY_EVEN | Select DMA parity Even |
| LL_SPI_DMA_PARITY_ODD  | Select DMA parity Odd  |

**Get Flags Defines**

|                  |                          |
|------------------|--------------------------|
| LL_SPI_SR_RXNE   | Rx buffer not empty flag |
| LL_SPI_SR_TXE    | Tx buffer empty flag     |
| LL_SPI_SR_BSY    | Busy flag                |
| LL_SPI_SR_CRCERR | CRC error flag           |
| LL_SPI_SR_MODF   | Mode fault flag          |

|                                          |                                                                                |
|------------------------------------------|--------------------------------------------------------------------------------|
| <code>LL_SPI_SR_OVR</code>               | Overrun flag                                                                   |
| <code>LL_SPI_SR_FRE</code>               | TI mode frame format error flag                                                |
| <b><i>IT Defines</i></b>                 |                                                                                |
| <code>LL_SPI_CR2_RXNEIE</code>           | Rx buffer not empty interrupt enable                                           |
| <code>LL_SPI_CR2_TXEIE</code>            | Tx buffer empty interrupt enable                                               |
| <code>LL_SPI_CR2_ERRIE</code>            | Error interrupt enable                                                         |
| <b><i>Operation Mode</i></b>             |                                                                                |
| <code>LL_SPI_MODE_MASTER</code>          | Master configuration                                                           |
| <code>LL_SPI_MODE_SLAVE</code>           | Slave configuration                                                            |
| <b><i>Slave Select Pin Mode</i></b>      |                                                                                |
| <code>LL_SPI_NSS_SOFT</code>             | NSS managed internally. NSS pin not used and free                              |
| <code>LL_SPI_NSS_HARD_INPUT</code>       | NSS pin used in Input. Only used in Master mode                                |
| <code>LL_SPI_NSS_HARD_OUTPUT</code>      | NSS pin used in Output. Only used in Slave mode as chip select                 |
| <b><i>Clock Phase</i></b>                |                                                                                |
| <code>LL_SPI_PHASE_1EDGE</code>          | First clock transition is the first data capture edge                          |
| <code>LL_SPI_PHASE_2EDGE</code>          | Second clock transition is the first data capture edge                         |
| <b><i>Clock Polarity</i></b>             |                                                                                |
| <code>LL_SPI_POLARITY_LOW</code>         | Clock to 0 when idle                                                           |
| <code>LL_SPI_POLARITY_HIGH</code>        | Clock to 1 when idle                                                           |
| <b><i>Serial Protocol</i></b>            |                                                                                |
| <code>LL_SPI_PROTOCOL_MOTOROLA</code>    | Motorola mode. Used as default value                                           |
| <code>LL_SPI_PROTOCOL_TI</code>          | TI mode                                                                        |
| <b><i>RX FIFO Level</i></b>              |                                                                                |
| <code>LL_SPI_RX_FIFO_EMPTY</code>        | FIFO reception empty                                                           |
| <code>LL_SPI_RX_FIFO_QUARTER_FULL</code> | FIFO reception 1/4                                                             |
| <code>LL_SPI_RX_FIFO_HALF_FULL</code>    | FIFO reception 1/2                                                             |
| <code>LL_SPI_RX_FIFO_FULL</code>         | FIFO reception full                                                            |
| <b><i>RX FIFO Threshold</i></b>          |                                                                                |
| <code>LL_SPI_RX_FIFO_TH_HALF</code>      | RXNE event is generated if FIFO level is greater than or equal to 1/2 (16-bit) |
| <code>LL_SPI_RX_FIFO_TH_QUARTER</code>   | RXNE event is generated if FIFO level is greater than or equal to 1/4 (8-bit)  |
| <b><i>Transfer Mode</i></b>              |                                                                                |
| <code>LL_SPI_FULL_DUPLEX</code>          | Full-Duplex mode. Rx and Tx transfer on 2 lines                                |
| <code>LL_SPI_SIMPLEX_RX</code>           | Simplex Rx mode. Rx transfer only on 1 line                                    |
| <code>LL_SPI_HALF_DUPLEX_RX</code>       | Half-Duplex Rx mode. Rx transfer on 1 line                                     |

`LL_SPI_HALF_DUPLEX_TX` Half-Duplex Tx mode. Tx transfer on 1 line

**TX FIFO Level**

`LL_SPI_TX_FIFO_EMPTY` FIFO transmission empty

`LL_SPI_TX_FIFO_QUARTER_FULL` FIFO transmission 1/4

`LL_SPI_TX_FIFO_HALF_FULL` FIFO transmission 1/2

`LL_SPI_TX_FIFO_FULL` FIFO transmission full

**Common Write and read registers Macros**

`LL_SPI_WriteReg` **Description:**

- Write a value in SPI register.

**Parameters:**

- `__INSTANCE__`: SPI Instance
- `__REG__`: Register to be written
- `__VALUE__`: Value to be written in the register

**Return value:**

- None

`LL_SPI_ReadReg`

**Description:**

- Read a value in SPI register.

**Parameters:**

- `__INSTANCE__`: SPI Instance
- `__REG__`: Register to be read

**Return value:**

- Register: value

## 77 LL SYSTEM Generic Driver

### 77.1 SYSTEM Firmware driver API description

#### 77.1.1 Detailed description of functions

##### **LL\_SYSCFG\_SetRemapMemory**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSCFG_SetRemapMemory(<br/>  uint32_t Memory)</code>                                                                                                                                                                                                                                                                                                                                 |
| Function description                              | Set memory mapping at address 0x00000000.                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Memory:</b> This parameter can be one of the following values:<br/>(*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- <code>LL_SYSCFG_REMAP_FLASH</code></li> <li>- <code>LL_SYSCFG_REMAP_SYSTEMFLASH</code></li> <li>- <code>LL_SYSCFG_REMAP_SRAM</code></li> <li>- <code>LL_SYSCFG_REMAP_FMC</code> (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>SYSCFG_CFGR1 MEM_MODE</code></li> <li>• <code>LL_SYSCFG_SetRemapMemory</code></li> </ul>                                                                                                                                                                                                                                                                            |

##### **LL\_SYSCFG\_GetRemapMemory**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SYSCFG_GetRemapMemory(<br/>  void )</code>                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Get memory mapping at address 0x00000000.                                                                                                                                                                                                                                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (*)<br/>value not defined in all devices.           <ul style="list-style-type: none"> <li>- <code>LL_SYSCFG_REMAP_FLASH</code></li> <li>- <code>LL_SYSCFG_REMAP_SYSTEMFLASH</code></li> <li>- <code>LL_SYSCFG_REMAP_SRAM</code></li> <li>- <code>LL_SYSCFG_REMAP_FMC</code> (*)</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>SYSCFG_CFGR1 MEM_MODE</code></li> <li>• <code>LL_SYSCFG_GetRemapMemory</code></li> </ul>                                                                                                                                                                                                                                                                     |

##### **LL\_SYSCFG\_SetRemapDMA\_ADC**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_SYSCFG_SetRemapDMA_ADC(<br/>  uint32_t Remap)</code>                                                                                                                                                                                                                                                                                                             |
| Function description | Set DMA request remapping bits for ADC.                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Remap:</b> This parameter can be one of the following values:<br/>(*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- <code>LL_SYSCFG_ADC24_RMP_DMA2_CH12</code> (*)</li> <li>- <code>LL_SYSCFG_ADC24_RMP_DMA2_CH34</code> (*)</li> <li>- <code>LL_SYSCFG_ADC2_RMP_DMA1_CH2</code> (*)</li> </ul> </li> </ul> |

- LL\_SYSCFG\_ADC2\_RMP\_DMA1\_CH4 (\*)
- LL\_SYSCFG\_ADC2\_RMP\_DMA2 (\*)
- LL\_SYSCFG\_ADC2\_RMP\_DMA1 (\*)

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- SYSCFG\_CFGR1 ADC24\_DMA\_RMP  
LL\_SYSCFG\_SetRemapDMA\_ADC
- SYSCFG\_CFGR3 ADC2\_DMA\_RMP  
LL\_SYSCFG\_SetRemapDMA\_ADC

### **LL\_SYSCFG\_SetRemapDMA\_DAC**

Function name      **\_\_STATIC\_INLINE void LL\_SYSCFG\_SetRemapDMA\_DAC  
(uint32\_t Remap)**

Function description      Set DMA request remapping bits for DAC.

Parameters

- **Remap:** This parameter can be one of the following values:  
(\*) value not defined in all devices.
  - LL\_SYSCFG\_DAC1\_CH1\_RMP\_DMA2\_CH3
  - LL\_SYSCFG\_DAC1\_CH1\_RMP\_DMA1\_CH3
  - LL\_SYSCFG\_DAC1\_OUT2\_RMP\_DMA2\_CH4 (\*)
  - LL\_SYSCFG\_DAC1\_OUT2\_RMP\_DMA1\_CH4 (\*)
  - LL\_SYSCFG\_DAC2\_OUT1\_RMP\_DMA2\_CH5 (\*)
  - LL\_SYSCFG\_DAC2\_OUT1\_RMP\_DMA1\_CH5 (\*)
  - LL\_SYSCFG\_DAC2\_CH1\_RMP\_NO (\*)
  - LL\_SYSCFG\_DAC2\_CH1\_RMP\_DMA1\_CH5 (\*)

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- SYSCFG\_CFGR1 TIM6DAC1Ch1\_DMA\_RMP  
LL\_SYSCFG\_SetRemapDMA\_DAC
- SYSCFG\_CFGR1 DAC2Ch1\_DMA\_RMP  
LL\_SYSCFG\_SetRemapDMA\_DAC

### **LL\_SYSCFG\_SetRemapDMA\_TIM**

Function name      **\_\_STATIC\_INLINE void LL\_SYSCFG\_SetRemapDMA\_TIM  
(uint32\_t Remap)**

Function description      Set DMA request remapping bits for TIM.

Parameters

- **Remap:** This parameter can be a combination of the following values: (\*) value not defined in all devices.
  - LL\_SYSCFG\_TIM16\_RMP\_DMA1\_CH3 or  
LL\_SYSCFG\_TIM16\_RMP\_DMA1\_CH6
  - LL\_SYSCFG\_TIM17\_RMP\_DMA1\_CH1 or  
LL\_SYSCFG\_TIM17\_RMP\_DMA1\_CH7
  - LL\_SYSCFG\_TIM6\_RMP\_DMA2\_CH3 or  
LL\_SYSCFG\_TIM6\_RMP\_DMA1\_CH3
  - LL\_SYSCFG\_TIM7\_RMP\_DMA2\_CH4 or  
LL\_SYSCFG\_TIM7\_RMP\_DMA1\_CH4 (\*)
  - LL\_SYSCFG\_TIM18\_RMP\_DMA2\_CH5 or  
LL\_SYSCFG\_TIM18\_RMP\_DMA1\_CH5 (\*)

Return values

- **None**

- Reference Manual to LL API cross reference:
- SYSCFG\_CFGR1\_TIM16\_DMA\_RMP  
LL\_SYSCFG\_SetRemapDMA\_TIM
  - SYSCFG\_CFGR1\_TIM17\_DMA\_RMP  
LL\_SYSCFG\_SetRemapDMA\_TIM
  - SYSCFG\_CFGR1\_TIM6DAC1Ch1\_DMA\_RMP  
LL\_SYSCFG\_SetRemapDMA\_TIM
  - SYSCFG\_CFGR1\_TIM7DAC1Ch2\_DMA\_RMP  
LL\_SYSCFG\_SetRemapDMA\_TIM
  - SYSCFG\_CFGR1\_TIM18DAC2Ch1\_DMA\_RMP  
LL\_SYSCFG\_SetRemapDMA\_TIM

### **LL\_SYSCFG\_SetRemapInput\_TIM**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_SYSCFG_SetRemapInput_TIM (uint32_t Remap)</code>                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description                        | Set Timer input remap.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>Remap:</b> This parameter can be one of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- LL_SYSCFG_TIM1_ITR3_RMP_TIM4_TRGO (*)</li> <li>- LL_SYSCFG_TIM1_ITR3_RMP_TIM17_OC (*)</li> <li>- LL_SYSCFG_TIM15_ENCODEMODE_NOREDIRECTION (*)</li> <li>- LL_SYSCFG_TIM15_ENCODEMODE_TIM2 (*)</li> <li>- LL_SYSCFG_TIM15_ENCODEMODE_TIM3 (*)</li> <li>- LL_SYSCFG_TIM15_ENCODEMODE_TIM4 (*)</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1_TIM1_ITR3_RMP<br/>LL_SYSCFG_SetRemapInput_TIM</li> <li>• SYSCFG_CFGR1_ENCODER_MODE<br/>LL_SYSCFG_SetRemapInput_TIM</li> </ul>                                                                                                                                                                                                                                                                                                                        |

### **LL\_SYSCFG\_SetRemapTrigger\_DAC**

|                                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                    | <code>__STATIC_INLINE void LL_SYSCFG_SetRemapTrigger_DAC (uint32_t Remap)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description             | Set DAC Trigger remap.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                       | <ul style="list-style-type: none"> <li>• <b>Remap:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_SYSCFG_DAC1_TRIG1_RMP_TIM8_TRGO (*)</li> <li>- LL_SYSCFG_DAC1_TRIG1_RMP_TIM3_TRGO (*)</li> <li>- LL_SYSCFG_DAC1_TRIG3_RMP_TIM15_TRGO (*)</li> <li>- LL_SYSCFG_DAC1_TRIG3_RMP_HRTIM1_DAC1_TRIG1 (*)</li> <li>- LL_SYSCFG_DAC1_TRIG5_RMP_NO (*)</li> <li>- LL_SYSCFG_DAC1_TRIG5_RMP_HRTIM1_DAC1_TRIG2 (*)</li> </ul> (*) value not defined in all devices. </li> </ul> |
| Return values                    | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to LL API cross | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1_DAC1_TRIG1_RMP<br/>LL_SYSCFG_SetRemapTrigger_DAC</li> <li>• SYSCFG_CFGR3_DAC1_TRIG3_RMP</li> </ul>                                                                                                                                                                                                                                                                                                                                                                   |

- reference:
- LL\_SYSCFG\_SetRemapTrigger\_DAC
  - SYSCFG\_CFGR3 DAC1\_TRG5\_RMP
  - LL\_SYSCFG\_SetRemapTrigger\_DAC

### **LL\_SYSCFG\_EnableRemapIT\_USB**

|                                                   |                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_SYSCFG_EnableRemapIT_USB(void)</code></b>                             |
| Function description                              | Enable USB interrupt remap.                                                                            |
| Return values                                     | • <b>None</b>                                                                                          |
| Notes                                             | • Remap the USB interrupts (USB_HP, USB_LP and USB_WKUP) on interrupt lines 74, 75 and 76 respectively |
| Reference Manual to<br>LL API cross<br>reference: | • SYSCFG_CFGR1 USB_IT_RMP<br>LL_SYSCFG_EnableRemapIT_USB                                               |

### **LL\_SYSCFG\_DisableRemapIT\_USB**

|                                                   |                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_SYSCFG_DisableRemapIT_USB(void)</code></b> |
| Function description                              | Disable USB interrupt remap.                                                |
| Return values                                     | • <b>None</b>                                                               |
| Reference Manual to<br>LL API cross<br>reference: | • SYSCFG_CFGR1 USB_IT_RMP<br>LL_SYSCFG_DisableRemapIT_USB                   |

### **LL\_SYSCFG\_EnableFastModePlus**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_SYSCFG_EnableFastModePlus(uint32_t ConfigFastModePlus)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Enable the I2C fast mode plus driving capability.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>ConfigFastModePlus:</b> This parameter can be a combination of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- LL_SYSCFG_I2C_FASTMODEPLUS_PB6</li> <li>- LL_SYSCFG_I2C_FASTMODEPLUS_PB7</li> <li>- LL_SYSCFG_I2C_FASTMODEPLUS_PB8</li> <li>- LL_SYSCFG_I2C_FASTMODEPLUS_PB9</li> <li>- LL_SYSCFG_I2C_FASTMODEPLUS_I2C1</li> <li>- LL_SYSCFG_I2C_FASTMODEPLUS_I2C2 (*)</li> <li>- LL_SYSCFG_I2C_FASTMODEPLUS_I2C3 (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1 I2C_PB6_FMP<br/>LL_SYSCFG_EnableFastModePlus</li> <li>• SYSCFG_CFGR1 I2C_PB7_FMP<br/>LL_SYSCFG_EnableFastModePlus</li> <li>• SYSCFG_CFGR1 I2C_PB8_FMP<br/>LL_SYSCFG_EnableFastModePlus</li> <li>• SYSCFG_CFGR1 I2C_PB9_FMP</li> </ul>                                                                                                                                                                                                                                                  |

- LL\_SYSCFG\_EnableFastModePlus
- SYSCFG\_CFGR1\_I2C1\_FMP
- LL\_SYSCFG\_EnableFastModePlus
- SYSCFG\_CFGR1\_I2C2\_FMP
- LL\_SYSCFG\_EnableFastModePlus
- SYSCFG\_CFGR1\_I2C3\_FMP
- LL\_SYSCFG\_EnableFastModePlus

### **LL\_SYSCFG\_DisableFastModePlus**

**Function name** `_STATIC_INLINE void LL_SYSCFG_DisableFastModePlus  
(uint32_t ConfigFastModePlus)`

**Function description** Disable the I2C fast mode plus driving capability.

- Parameters**
- **ConfigFastModePlus:** This parameter can be a combination of the following values: (\*) value not defined in all devices.
    - LL\_SYSCFG\_I2C\_FASTMODEPLUS\_PB6
    - LL\_SYSCFG\_I2C\_FASTMODEPLUS\_PB7
    - LL\_SYSCFG\_I2C\_FASTMODEPLUS\_PB8
    - LL\_SYSCFG\_I2C\_FASTMODEPLUS\_PB9
    - LL\_SYSCFG\_I2C\_FASTMODEPLUS\_I2C1
    - LL\_SYSCFG\_I2C\_FASTMODEPLUS\_I2C2 (\*)
    - LL\_SYSCFG\_I2C\_FASTMODEPLUS\_I2C3 (\*)

- Return values**
- **None**

- Reference Manual to LL API cross reference:**
- SYSCFG\_CFGR1\_I2C\_PB6\_FMP
  - LL\_SYSCFG\_DisableFastModePlus
  - SYSCFG\_CFGR1\_I2C\_PB7\_FMP
  - LL\_SYSCFG\_DisableFastModePlus
  - SYSCFG\_CFGR1\_I2C\_PB8\_FMP
  - LL\_SYSCFG\_DisableFastModePlus
  - SYSCFG\_CFGR1\_I2C\_PB9\_FMP
  - LL\_SYSCFG\_DisableFastModePlus
  - SYSCFG\_CFGR1\_I2C1\_FMP
  - LL\_SYSCFG\_DisableFastModePlus
  - SYSCFG\_CFGR1\_I2C2\_FMP
  - LL\_SYSCFG\_DisableFastModePlus
  - SYSCFG\_CFGR1\_I2C3\_FMP
  - LL\_SYSCFG\_DisableFastModePlus

### **LL\_SYSCFG\_EnableIT\_FPU\_IOC**

**Function name** `_STATIC_INLINE void LL_SYSCFG_EnableIT_FPU_IOC (void )`

**Function description** Enable Floating Point Unit Invalid operation Interrupt.

- Return values**
- **None**

- Reference Manual to LL API cross reference:**
- SYSCFG\_CFGR1\_FPU\_IE\_0
  - LL\_SYSCFG\_EnableIT\_FPU\_IOC

**LL\_SYSCFG\_EnableIT\_FPU\_DZC**

|                                                   |                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSCFG_EnableIT_FPU_DZC (void )</code>                                            |
| Function description                              | Enable Floating Point Unit Divide-by-zero Interrupt.                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1 FPU_IE_1</li> <li>• LL_SYSCFG_EnableIT_FPU_DZC</li> </ul> |

**LL\_SYSCFG\_EnableIT\_FPU\_UFC**

|                                                   |                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSCFG_EnableIT_FPU_UFC (void )</code>                                            |
| Function description                              | Enable Floating Point Unit Underflow Interrupt.                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1 FPU_IE_2</li> <li>• LL_SYSCFG_EnableIT_FPU_UFC</li> </ul> |

**LL\_SYSCFG\_EnableIT\_FPU\_OFC**

|                                                   |                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSCFG_EnableIT_FPU_OFC (void )</code>                                            |
| Function description                              | Enable Floating Point Unit Overflow Interrupt.                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1 FPU_IE_3</li> <li>• LL_SYSCFG_EnableIT_FPU_OFC</li> </ul> |

**LL\_SYSCFG\_EnableIT\_FPU\_IDC**

|                                                   |                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSCFG_EnableIT_FPU_IDC (void )</code>                                            |
| Function description                              | Enable Floating Point Unit Input denormal Interrupt.                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1 FPU_IE_4</li> <li>• LL_SYSCFG_EnableIT_FPU_IDC</li> </ul> |

**LL\_SYSCFG\_EnableIT\_FPU\_IXC**

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_SYSCFG_EnableIT_FPU_IXC (void )</code>      |
| Function description | Enable Floating Point Unit Inexact Interrupt.                             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>           |
| Reference Manual to  | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1 FPU_IE_5</li> </ul> |

LL API cross reference:  
LL\_SYSCFG\_EnableIT\_FPU\_IXC

### **LL\_SYSCFG\_DisableIT\_FPU\_IOC**

Function name      **\_\_STATIC\_INLINE void LL\_SYSCFG\_DisableIT\_FPU\_IOC (void )**

Function description      Disable Floating Point Unit Invalid operation Interrupt.

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
SYSCFG\_CFGR1 FPU\_IE\_0  
LL\_SYSCFG\_DisableIT\_FPU\_IOC

### **LL\_SYSCFG\_DisableIT\_FPU\_DZC**

Function name      **\_\_STATIC\_INLINE void LL\_SYSCFG\_DisableIT\_FPU\_DZC (void )**

Function description      Disable Floating Point Unit Divide-by-zero Interrupt.

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
SYSCFG\_CFGR1 FPU\_IE\_1  
LL\_SYSCFG\_DisableIT\_FPU\_DZC

### **LL\_SYSCFG\_DisableIT\_FPU\_UFC**

Function name      **\_\_STATIC\_INLINE void LL\_SYSCFG\_DisableIT\_FPU\_UFC (void )**

Function description      Disable Floating Point Unit Underflow Interrupt.

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
SYSCFG\_CFGR1 FPU\_IE\_2  
LL\_SYSCFG\_DisableIT\_FPU\_UFC

### **LL\_SYSCFG\_DisableIT\_FPU\_OFC**

Function name      **\_\_STATIC\_INLINE void LL\_SYSCFG\_DisableIT\_FPU\_OFC (void )**

Function description      Disable Floating Point Unit Overflow Interrupt.

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
SYSCFG\_CFGR1 FPU\_IE\_3  
LL\_SYSCFG\_DisableIT\_FPU\_OFC

**LL\_SYSCFG\_DisableIT\_FPU\_IDC**

|                                                   |                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSCFG_DisableIT_FPU_IDC (void )</code>                                                  |
| Function description                              | Disable Floating Point Unit Input denormal Interrupt.                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1 FPU_IE_4<br/><code>LL_SYSCFG_DisableIT_FPU_IDC</code></li> </ul> |

**LL\_SYSCFG\_DisableIT\_FPU\_IXC**

|                                                   |                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSCFG_DisableIT_FPU_IXC (void )</code>                                                  |
| Function description                              | Disable Floating Point Unit Inexact Interrupt.                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1 FPU_IE_5<br/><code>LL_SYSCFG_DisableIT_FPU_IXC</code></li> </ul> |

**LL\_SYSCFG\_IsEnabledIT\_FPU\_IOC**

|                                                   |                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SYSCFG_IsEnabledIT_FPU_IOC (void )</code>                                              |
| Function description                              | Check if Floating Point Unit Invalid operation Interrupt source is enabled or disabled.                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1 FPU_IE_0<br/><code>LL_SYSCFG_IsEnabledIT_FPU_IOC</code></li> </ul> |

**LL\_SYSCFG\_IsEnabledIT\_FPU\_DZC**

|                                                   |                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SYSCFG_IsEnabledIT_FPU_DZC (void )</code>                                              |
| Function description                              | Check if Floating Point Unit Divide-by-zero Interrupt source is enabled or disabled.                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SYSCFG_CFGR1 FPU_IE_1<br/><code>LL_SYSCFG_IsEnabledIT_FPU_DZC</code></li> </ul> |

**LL\_SYSCFG\_IsEnabledIT\_FPU\_UFC**

|                      |                                                                                 |
|----------------------|---------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_SYSCFG_IsEnabledIT_FPU_UFC (void )</code>     |
| Function description | Check if Floating Point Unit Underflow Interrupt source is enabled or disabled. |

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SYSCFG_CFGR1 FPU_IE_2<br/>LL_SYSCFG_IsEnabledIT_FPU_UFC</li> </ul> |

### LL\_SYSCFG\_IsEnabledIT\_FPU\_OFC

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_SYSCFG_IsEnabledIT_FPU_OFC (void )</code>                           |
| Function description                              | Check if Floating Point Unit Overflow Interrupt source is enabled or disabled.                            |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SYSCFG_CFGR1 FPU_IE_3<br/>LL_SYSCFG_IsEnabledIT_FPU_OFC</li> </ul> |

### LL\_SYSCFG\_IsEnabledIT\_FPU\_IDC

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_SYSCFG_IsEnabledIT_FPU_IDC (void )</code>                           |
| Function description                              | Check if Floating Point Unit Input denormal Interrupt source is enabled or disabled.                      |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SYSCFG_CFGR1 FPU_IE_4<br/>LL_SYSCFG_IsEnabledIT_FPU_IDC</li> </ul> |

### LL\_SYSCFG\_IsEnabledIT\_FPU\_IXC

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_SYSCFG_IsEnabledIT_FPU_IXC (void )</code>                           |
| Function description                              | Check if Floating Point Unit Inexact Interrupt source is enabled or disabled.                             |
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>SYSCFG_CFGR1 FPU_IE_5<br/>LL_SYSCFG_IsEnabledIT_FPU_IXC</li> </ul> |

### LL\_SYSCFG\_SetEXTISource

|                      |                                                                                                                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_SYSCFG_SetEXTISource (uint32_t<br/>Port, uint32_t Line)</code>                                                                                                                                                                                                |
| Function description | Configure source input for the EXTI external interrupt.                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>Port:</b> This parameter can be one of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>LL_SYSCFG_EXTI_PORTA</li> <li>LL_SYSCFG_EXTI_PORTB</li> <li>LL_SYSCFG_EXTI_PORTC</li> </ul> </li> </ul> |

- LL\_SYSCFG\_EXTI\_PORTD
- LL\_SYSCFG\_EXTI\_PORTE (\*)
- LL\_SYSCFG\_EXTI\_PORTF
- LL\_SYSCFG\_EXTI\_PORTG (\*)
- LL\_SYSCFG\_EXTI\_PORTH (\*)
- **Line:** This parameter can be one of the following values:
  - LL\_SYSCFG\_EXTI\_LINE0
  - LL\_SYSCFG\_EXTI\_LINE1
  - LL\_SYSCFG\_EXTI\_LINE2
  - LL\_SYSCFG\_EXTI\_LINE3
  - LL\_SYSCFG\_EXTI\_LINE4
  - LL\_SYSCFG\_EXTI\_LINE5
  - LL\_SYSCFG\_EXTI\_LINE6
  - LL\_SYSCFG\_EXTI\_LINE7
  - LL\_SYSCFG\_EXTI\_LINE8
  - LL\_SYSCFG\_EXTI\_LINE9
  - LL\_SYSCFG\_EXTI\_LINE10
  - LL\_SYSCFG\_EXTI\_LINE11
  - LL\_SYSCFG\_EXTI\_LINE12
  - LL\_SYSCFG\_EXTI\_LINE13
  - LL\_SYSCFG\_EXTI\_LINE14
  - LL\_SYSCFG\_EXTI\_LINE15
- **None**
- Reference Manual to LL API cross reference:
- SYSCFG\_EXTICR1 EXTI0 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI1 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI2 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI3 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI4 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI5 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI6 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI7 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI8 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI9 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI10 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI11 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI12 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI13 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI14 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR1 EXTI15 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI0 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI1 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI2 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI3 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI4 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI5 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI6 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI7 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI8 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI9 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI10 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI11 LL\_SYSCFG\_SetEXTISource

- SYSCFG\_EXTICR2 EXTI12 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI13 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI14 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR2 EXTI15 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI0 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI1 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI2 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI3 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI4 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI5 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI6 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI7 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI8 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI9 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI10 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI11 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI12 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI13 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI14 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR3 EXTI15 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI0 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI1 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI2 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI3 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI4 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI5 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI6 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI7 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI8 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI9 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI10 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI11 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI12 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI13 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI14 LL\_SYSCFG\_SetEXTISource
- SYSCFG\_EXTICR4 EXTI15 LL\_SYSCFG\_SetEXTISource

### LL\_SYSCFG\_GetEXTISource

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_SYSCFG_GetEXTISource<br/>(uint32_t Line)</code>                                                                                                                                                                                                                                                                                                                                                        |
| Function description | Get the configured defined for specific EXTI Line.                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Line:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_SYSCFG_EXTI_LINE0</li> <li>- LL_SYSCFG_EXTI_LINE1</li> <li>- LL_SYSCFG_EXTI_LINE2</li> <li>- LL_SYSCFG_EXTI_LINE3</li> <li>- LL_SYSCFG_EXTI_LINE4</li> <li>- LL_SYSCFG_EXTI_LINE5</li> <li>- LL_SYSCFG_EXTI_LINE6</li> <li>- LL_SYSCFG_EXTI_LINE7</li> </ul> </li> </ul> |

- LL\_SYSCFG\_EXTI\_LINE8
- LL\_SYSCFG\_EXTI\_LINE9
- LL\_SYSCFG\_EXTI\_LINE10
- LL\_SYSCFG\_EXTI\_LINE11
- LL\_SYSCFG\_EXTI\_LINE12
- LL\_SYSCFG\_EXTI\_LINE13
- LL\_SYSCFG\_EXTI\_LINE14
- LL\_SYSCFG\_EXTI\_LINE15

**Return values**

- **Returned:** value can be one of the following values: (\*)  
value not defined in all devices.
  - LL\_SYSCFG\_EXTI\_PORTA
  - LL\_SYSCFG\_EXTI\_PORTB
  - LL\_SYSCFG\_EXTI\_PORTC
  - LL\_SYSCFG\_EXTI\_PORTD
  - LL\_SYSCFG\_EXTI PORTE (\*)
  - LL\_SYSCFG\_EXTI\_PORTF
  - LL\_SYSCFG\_EXTI\_PORTG (\*)
  - LL\_SYSCFG\_EXTI\_PORH (\*)

**Reference Manual to  
LL API cross  
reference:**

- SYSCFG\_EXTICR1 EXTI0 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI1 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI2 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI3 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI4 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI5 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI6 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI7 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI8 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI9 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI10 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI11 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI12 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI13 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI14 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR1 EXTI15 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI0 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI1 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI2 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI3 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI4 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI5 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI6 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI7 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI8 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI9 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI10 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI11 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI12 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI13 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI14 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR2 EXTI15 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI0 LL\_SYSCFG\_GetEXTISource

- SYSCFG\_EXTICR3 EXTI1 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI2 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI3 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI4 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI5 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI6 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI7 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI8 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI9 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI10 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI11 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI12 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI13 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI14 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR3 EXTI15 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI0 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI1 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI2 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI3 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI4 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI5 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI6 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI7 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI8 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI9 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI10 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI11 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI12 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI13 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI14 LL\_SYSCFG\_GetEXTISource
- SYSCFG\_EXTICR4 EXTI15 LL\_SYSCFG\_GetEXTISource

### **LL\_SYSCFG\_SetTIMBreakInputs**

**Function name** `__STATIC_INLINE void LL_SYSCFG_SetTIMBreakInputs(  
                  uint32_t Break)`

**Function description** Set connections to TIMx Break inputs.

**Parameters**

- **Break:** This parameter can be a combination of the following values: (\*) value not defined in all devices.
  - LL\_SYSCFG\_TIMBREAK\_PVD (\*)
  - LL\_SYSCFG\_TIMBREAK\_SRAM\_PARITY (\*)
  - LL\_SYSCFG\_TIMBREAK\_LOCKUP

**Return values**

**Reference Manual to  
LL API cross  
reference:**

- SYSCFG\_CFGR2 LOCKUP\_LOCK  
LL\_SYSCFG\_SetTIMBreakInputs
- SYSCFG\_CFGR2 SRAM\_PARITY\_LOCK  
LL\_SYSCFG\_SetTIMBreakInputs
- SYSCFG\_CFGR2 PVD\_LOCK  
LL\_SYSCFG\_SetTIMBreakInputs

**LL\_SYSCFG\_GetTIMBreakInputs**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SYSCFG_GetTIMBreakInputs(void)</code>                                                                                                                                                                                                                                                                                                              |
| Function description                              | Get connections to TIMx Break inputs.                                                                                                                                                                                                                                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be a combination of the following values: (*) value not defined in all devices.             <ul style="list-style-type: none"> <li>- <code>LL_SYSCFG_TIMBREAK_PVD</code> (*)</li> <li>- <code>LL_SYSCFG_TIMBREAK_SRAM_PARITY</code> (*)</li> <li>- <code>LL_SYSCFG_TIMBREAK_LOCKUP</code></li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>SYSCFG_CFGR2_LOCKUP_LOCK</code><br/><code>LL_SYSCFG_GetTIMBreakInputs</code></li> <li>• <code>SYSCFG_CFGR2_SRAM_PARITY_LOCK</code><br/><code>LL_SYSCFG_GetTIMBreakInputs</code></li> <li>• <code>SYSCFG_CFGR2_PVD_LOCK</code><br/><code>LL_SYSCFG_GetTIMBreakInputs</code></li> </ul>                                                 |

**LL\_SYSCFG\_DisableSRAMParityCheck**

|                                                   |                                                                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSCFG_DisableSRAMParityCheck(void)</code>                                                                     |
| Function description                              | Disable RAM Parity Check.                                                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>SYSCFG_CFGR2_BYP_ADDR_PAR</code><br/><code>LL_SYSCFG_DisableSRAMParityCheck</code></li> </ul> |

**LL\_SYSCFG\_IsActiveFlag\_SP**

|                                                   |                                                                                                                              |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_SYSCFG_IsActiveFlag_SP(void)</code>                                                        |
| Function description                              | Check if SRAM parity error detected.                                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>SYSCFG_CFGR2_SRAM_PE</code> <code>LL_SYSCFG_IsActiveFlag_SP</code></li> </ul> |

**LL\_SYSCFG\_ClearFlag\_SP**

|                                                   |                                                                                                                           |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_SYSCFG_ClearFlag_SP(void)</code>                                                            |
| Function description                              | Clear SRAM parity error flag.                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>SYSCFG_CFGR2_SRAM_PE</code> <code>LL_SYSCFG_ClearFlag_SP</code></li> </ul> |

**LL\_SYSCFG\_EnableCCM\_SRAMPageWRP**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP (uint32_t<br/>PageWRP)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Enable CCM SRAM page write protection.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>PageWRP:</b> This parameter can be a combination of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE0</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE1</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE2</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE3</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE4 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE5 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE6 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE7 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE8 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE9 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE10 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE11 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE12 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE13 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE14 (*)</li> <li>- LL_SYSCFG_CCMSRAMWRP_PAGE15 (*)</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• Write protection is cleared only by a system reset</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SYSCFG_RCR PAGE0<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE1<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE2<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE3<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE4<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE5<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE6<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE7<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE8<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE9<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE10<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE11<br/>LL_SYSCFG_EnableCCM_SRAMPageWRP</li> <li>• SYSCFG_RCR PAGE12</li> </ul>                                |

- LL\_SYSCFG\_EnableCCM\_SRAMPAGEWRP
- SYSCFG\_RCR PAGE13
- LL\_SYSCFG\_EnableCCM\_SRAMPAGEWRP
- SYSCFG\_RCR PAGE14
- LL\_SYSCFG\_EnableCCM\_SRAMPAGEWRP
- SYSCFG\_RCR PAGE15
- LL\_SYSCFG\_EnableCCM\_SRAMPAGEWRP

### **LL\_DBGMCU\_GetDeviceID**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_DBGMCU_GetDeviceID (void )</code></b>                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Return the device identifier.                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Values:</b> between Min_Data=0x00 and Max_Data=0xFFFF</li> </ul>                                                                                                                                                                                                                                                                                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• For STM32F303xC, STM32F358xx and STM32F302xC devices, the device ID is 0x422</li> <li>• For STM32F373xx and STM32F378xx devices, the device ID is 0x432</li> <li>• For STM32F303x8, STM32F334xx and STM32F328xx devices, the device ID is 0x438.</li> <li>• For STM32F302x8, STM32F301x8 and STM32F318xx devices, the device ID is 0x439</li> <li>• For STM32F303xE, STM32F398xx and STM32F302xE devices, the device ID is 0x446</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DBGMCU_IDCODE DEV_ID LL_DBGMCU_GetDeviceID</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                       |

### **LL\_DBGMCU\_GetRevisionID**

|                                                   |                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_DBGMCU_GetRevisionID (void )</code></b>                                 |
| Function description                              | Return the device revision identifier.                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Values:</b> between Min_Data=0x00 and Max_Data=0xFFFF</li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• This field indicates the revision of the device.</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DBGMCU_IDCODE REV_ID LL_DBGMCU_GetRevisionID</li> </ul>             |

### **LL\_DBGMCU\_EnableDBGSleepMode**

|                                                   |                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_DBGMCU_EnableDBGSleepMode (void )</code></b>                                   |
| Function description                              | Enable the Debug Module during SLEEP mode.                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DBGMCU_CR DBG_SLEEP</li> <li>• LL_DBGMCU_EnableDBGSleepMode</li> </ul> |

**LL\_DBGMCU\_DisableDBGSleepMode**

|                                                   |                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DBGMCU_DisableDBGSleepMode(void)</code>                                                                      |
| Function description                              | Disable the Debug Module during SLEEP mode.                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>DBGMCU_CR DBG_SLEEP</code></li> <li>• <code>LL_DBGMCU_DisableDBGSleepMode</code></li> </ul> |

**LL\_DBGMCU\_EnableDBGStopMode**

|                                                   |                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DBGMCU_EnableDBGStopMode(void)</code>                                                                     |
| Function description                              | Enable the Debug Module during STOP mode.                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>DBGMCU_CR DBG_STOP</code></li> <li>• <code>LL_DBGMCU_EnableDBGStopMode</code></li> </ul> |

**LL\_DBGMCU\_DisableDBGStopMode**

|                                                   |                                                                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DBGMCU_DisableDBGStopMode(void)</code>                                                                     |
| Function description                              | Disable the Debug Module during STOP mode.                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>DBGMCU_CR DBG_STOP</code></li> <li>• <code>LL_DBGMCU_DisableDBGStopMode</code></li> </ul> |

**LL\_DBGMCU\_EnableDBGStandbyMode**

|                                                   |                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_DBGMCU_EnableDBGStandbyMode(void)</code>                                                                        |
| Function description                              | Enable the Debug Module during STANDBY mode.                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>DBGMCU_CR DBG_STANDBY</code></li> <li>• <code>LL_DBGMCU_EnableDBGStandbyMode</code></li> </ul> |

**LL\_DBGMCU\_DisableDBGStandbyMode**

|                      |                                                                         |
|----------------------|-------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_DBGMCU_DisableDBGStandbyMode(void)</code> |
| Function description | Disable the Debug Module during STANDBY mode.                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>         |
| Reference Manual to  | <code>DBGMCU_CR DBG_STANDBY</code>                                      |

LL API cross reference:  
LL\_DBGMCU\_DisableDBGStandbyMode

### **LL\_DBGMCU\_SetTracePinAssignment**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_DBGMCU_SetTracePinAssignment (uint32_t PinAssignment)</code></b>                                                                                                                                                                                                                                                                                                                                         |
| Function description                              | Set Trace pin assignment control.                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>PinAssignment:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- <code>LL_DBGMCU_TRACE_NONE</code></li> <li>- <code>LL_DBGMCU_TRACE_ASYNCH</code></li> <li>- <code>LL_DBGMCU_TRACE_SYNCH_SIZE1</code></li> <li>- <code>LL_DBGMCU_TRACE_SYNCH_SIZE2</code></li> <li>- <code>LL_DBGMCU_TRACE_SYNCH_SIZE4</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>DBGMCU_CR TRACE_IOEN</code></li> <li>• <code>LL_DBGMCU_SetTracePinAssignment</code></li> <li>• <code>DBGMCU_CR TRACE_MODE</code></li> <li>• <code>LL_DBGMCU_SetTracePinAssignment</code></li> </ul>                                                                                                                                                                                       |

### **LL\_DBGMCU\_GetTracePinAssignment**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_DBGMCU_GetTracePinAssignment (void )</code></b>                                                                                                                                                                                                                                                                                                                                        |
| Function description                              | Get Trace pin assignment control.                                                                                                                                                                                                                                                                                                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- <code>LL_DBGMCU_TRACE_NONE</code></li> <li>- <code>LL_DBGMCU_TRACE_ASYNCH</code></li> <li>- <code>LL_DBGMCU_TRACE_SYNCH_SIZE1</code></li> <li>- <code>LL_DBGMCU_TRACE_SYNCH_SIZE2</code></li> <li>- <code>LL_DBGMCU_TRACE_SYNCH_SIZE4</code></li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>DBGMCU_CR TRACE_IOEN</code></li> <li>• <code>LL_DBGMCU_SetTracePinAssignment</code></li> <li>• <code>DBGMCU_CR TRACE_MODE</code></li> <li>• <code>LL_DBGMCU_SetTracePinAssignment</code></li> </ul>                                                                                                                                                                         |

### **LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_DBGMCU_APB1_GRP1_FreezePeriph (uint32_t Periph)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description | Freeze APB1 peripherals (group1 peripherals)                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices.           <ul style="list-style-type: none"> <li>- <code>LL_DBGMCU_APB1_GRP1_TIM2_STOP</code></li> <li>- <code>LL_DBGMCU_APB1_GRP1_TIM3_STOP</code> (*)</li> <li>- <code>LL_DBGMCU_APB1_GRP1_TIM4_STOP</code> (*)</li> <li>- <code>LL_DBGMCU_APB1_GRP1_TIM5_STOP</code> (*)</li> <li>- <code>LL_DBGMCU_APB1_GRP1_TIM6_STOP</code></li> </ul> </li> </ul> |

- LL\_DBGMCU\_APB1\_GRP1\_TIM7\_STOP (\*)
- LL\_DBGMCU\_APB1\_GRP1\_TIM12\_STOP (\*)
- LL\_DBGMCU\_APB1\_GRP1\_TIM13\_STOP (\*)
- LL\_DBGMCU\_APB1\_GRP1\_TIM14\_STOP (\*)
- LL\_DBGMCU\_APB1\_GRP1\_TIM18\_STOP (\*)
- LL\_DBGMCU\_APB1\_GRP1\_RTC\_STOP
- LL\_DBGMCU\_APB1\_GRP1\_WWDG\_STOP
- LL\_DBGMCU\_APB1\_GRP1\_IWDG\_STOP
- LL\_DBGMCU\_APB1\_GRP1\_I2C1\_STOP
- LL\_DBGMCU\_APB1\_GRP1\_I2C2\_STOP (\*)
- LL\_DBGMCU\_APB1\_GRP1\_I2C3\_STOP (\*)
- LL\_DBGMCU\_APB1\_GRP1\_CAN\_STOP (\*)

**Return values**

Reference Manual to  
LL API cross  
reference:

- **None**
- APB1\_FZ DBG\_TIM2\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_TIM3\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_TIM4\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_TIM5\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_TIM6\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_TIM7\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_TIM12\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_TIM13\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_TIM14\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_TIM18\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_RTC\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_WWDG\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_IWDG\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_I2C1\_SMBUS\_TIMEOUT  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_I2C2\_SMBUS\_TIMEOUT  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_I2C3\_SMBUS\_TIMEOUT  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph
- APB1\_FZ DBG\_CAN\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_FreezePeriph

**LL\_DBGMCU\_APB1\_GRP1\_UnFreezePeriph**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph (uint32_t<br/>Periph)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                              | Unfreeze APB1 peripherals (group1 peripherals)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>Periph:</b> This parameter can be a combination of the following values: (*) value not defined in all devices. <ul style="list-style-type: none"> <li>- LL_DBGMCU_APB1_GRP1_TIM2_STOP</li> <li>- LL_DBGMCU_APB1_GRP1_TIM3_STOP (*)</li> <li>- LL_DBGMCU_APB1_GRP1_TIM4_STOP (*)</li> <li>- LL_DBGMCU_APB1_GRP1_TIM5_STOP (*)</li> <li>- LL_DBGMCU_APB1_GRP1_TIM6_STOP</li> <li>- LL_DBGMCU_APB1_GRP1_TIM7_STOP (*)</li> <li>- LL_DBGMCU_APB1_GRP1_TIM12_STOP (*)</li> <li>- LL_DBGMCU_APB1_GRP1_TIM13_STOP (*)</li> <li>- LL_DBGMCU_APB1_GRP1_TIM14_STOP (*)</li> <li>- LL_DBGMCU_APB1_GRP1_TIM18_STOP (*)</li> <li>- LL_DBGMCU_APB1_GRP1_RTC_STOP</li> <li>- LL_DBGMCU_APB1_GRP1_WWDG_STOP</li> <li>- LL_DBGMCU_APB1_GRP1_IWDG_STOP</li> <li>- LL_DBGMCU_APB1_GRP1_I2C1_STOP</li> <li>- LL_DBGMCU_APB1_GRP1_I2C2_STOP (*)</li> <li>- LL_DBGMCU_APB1_GRP1_I2C3_STOP (*)</li> <li>- LL_DBGMCU_APB1_GRP1_CAN_STOP (*)</li> </ul> </li> </ul>                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• APB1_FZ DBG_TIM2_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_TIM3_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_TIM4_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_TIM5_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_TIM6_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_TIM7_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_TIM12_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_TIM13_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_TIM14_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_TIM18_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_RTC_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_WWDG_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> <li>• APB1_FZ DBG_IWDG_STOP<br/>LL_DBGMCU_APB1_GRP1_UnFreezePeriph</li> </ul> |

- APB1\_FZ DBG\_I2C1\_SMBUS\_TIMEOUT  
LL\_DBGMCU\_APB1\_GRP1\_UnFreezePeriph
- APB1\_FZ DBG\_I2C2\_SMBUS\_TIMEOUT  
LL\_DBGMCU\_APB1\_GRP1\_UnFreezePeriph
- APB1\_FZ DBG\_I2C3\_SMBUS\_TIMEOUT  
LL\_DBGMCU\_APB1\_GRP1\_UnFreezePeriph
- APB1\_FZ DBG\_CAN\_STOP  
LL\_DBGMCU\_APB1\_GRP1\_UnFreezePeriph

### **LL\_DBGMCU\_APB2\_GRP1\_FreezePeriph**

Function name      **\_\_STATIC\_INLINE void LL\_DBGMCU\_APB2\_GRP1\_FreezePeriph (uint32\_t Periph)**

Function description      Freeze APB2 peripherals.

- Parameters
- **Periph:** This parameter can be a combination of the following values: (\*) value not defined in all devices.
    - LL\_DBGMCU\_APB2\_GRP1\_TIM1\_STOP (\*)
    - LL\_DBGMCU\_APB2\_GRP1\_TIM8\_STOP (\*)
    - LL\_DBGMCU\_APB2\_GRP1\_TIM15\_STOP
    - LL\_DBGMCU\_APB2\_GRP1\_TIM16\_STOP
    - LL\_DBGMCU\_APB2\_GRP1\_TIM17\_STOP
    - LL\_DBGMCU\_APB2\_GRP1\_TIM19\_STOP (\*)
    - LL\_DBGMCU\_APB2\_GRP1\_TIM20\_STOP (\*)
    - LL\_DBGMCU\_APB2\_GRP1\_HRTIM1\_STOP (\*)

- Return values
- **None**

- Reference Manual to  
LL API cross  
reference:
- APB2\_FZ DBG\_TIM1\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_FreezePeriph
  - APB2\_FZ DBG\_TIM8\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_FreezePeriph
  - APB2\_FZ DBG\_TIM15\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_FreezePeriph
  - APB2\_FZ DBG\_TIM16\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_FreezePeriph
  - APB2\_FZ DBG\_TIM17\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_FreezePeriph
  - APB2\_FZ DBG\_TIM19\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_FreezePeriph
  - APB2\_FZ DBG\_TIM20\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_FreezePeriph
  - APB2\_FZ DBG\_HRTIM1\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_FreezePeriph

### **LL\_DBGMCU\_APB2\_GRP1\_UnFreezePeriph**

Function name      **\_\_STATIC\_INLINE void LL\_DBGMCU\_APB2\_GRP1\_UnFreezePeriph (uint32\_t Periph)**

Function description      Unfreeze APB2 peripherals.

- Parameters
- **Periph:** This parameter can be a combination of the following values: (\*) value not defined in all devices.
    - LL\_DBGMCU\_APB2\_GRP1\_TIM1\_STOP (\*)

- LL\_DBGMCU\_APB2\_GRP1\_TIM8\_STOP (\*)
- LL\_DBGMCU\_APB2\_GRP1\_TIM15\_STOP
- LL\_DBGMCU\_APB2\_GRP1\_TIM16\_STOP
- LL\_DBGMCU\_APB2\_GRP1\_TIM17\_STOP
- LL\_DBGMCU\_APB2\_GRP1\_TIM19\_STOP (\*)
- LL\_DBGMCU\_APB2\_GRP1\_TIM20\_STOP (\*)
- LL\_DBGMCU\_APB2\_GRP1\_HRTIM1\_STOP (\*)

**Return values**

- **None**

**Reference Manual to  
LL API cross  
reference:**

- APB2\_FZ DBG\_TIM1\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_UnFreezePeriph
- APB2\_FZ DBG\_TIM8\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_UnFreezePeriph
- APB2\_FZ DBG\_TIM15\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_UnFreezePeriph
- APB2\_FZ DBG\_TIM16\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_UnFreezePeriph
- APB2\_FZ DBG\_TIM17\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_UnFreezePeriph
- APB2\_FZ DBG\_TIM19\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_UnFreezePeriph
- APB2\_FZ DBG\_TIM20\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_UnFreezePeriph
- APB2\_FZ DBG\_HRTIM1\_STOP  
LL\_DBGMCU\_APB2\_GRP1\_UnFreezePeriph

### LL\_FLASH\_SetLatency

**Function name**      `__STATIC_INLINE void LL_FLASH_SetLatency (uint32_t Latency)`

**Function description**      Set FLASH Latency.

**Parameters**      • **Latency:** This parameter can be one of the following values:
 

- LL\_FLASH\_LATENCY\_0
- LL\_FLASH\_LATENCY\_1
- LL\_FLASH\_LATENCY\_2

**Return values**

- **None**

**Reference Manual to  
LL API cross  
reference:**

- FLASH\_ACR LATENCY LL\_FLASH\_SetLatency

### LL\_FLASH\_GetLatency

**Function name**      `__STATIC_INLINE uint32_t LL_FLASH_GetLatency (void )`

**Function description**      Get FLASH Latency.

**Return values**      • **Returned:** value can be one of the following values:
 

- LL\_FLASH\_LATENCY\_0
- LL\_FLASH\_LATENCY\_1
- LL\_FLASH\_LATENCY\_2

**Reference Manual to  
LL API cross**

- FLASH\_ACR LATENCY LL\_FLASH\_GetLatency

reference:

### **LL\_FLASH\_EnablePrefetch**

Function name      **\_\_STATIC\_INLINE void LL\_FLASH\_EnablePrefetch (void )**

Function description      Enable Prefetch.

Return values      •    **None**

Reference Manual to  
LL API cross  
reference:

### **LL\_FLASH\_DisablePrefetch**

Function name      **\_\_STATIC\_INLINE void LL\_FLASH\_DisablePrefetch (void )**

Function description      Disable Prefetch.

Return values      •    **None**

Reference Manual to  
LL API cross  
reference:

### **LL\_FLASH\_IsPrefetchEnabled**

Function name      **\_\_STATIC\_INLINE uint32\_t LL\_FLASH\_IsPrefetchEnabled  
(void )**

Function description      Check if Prefetch buffer is enabled.

Return values      •    **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

### **LL\_FLASH\_EnableHalfCycleAccess**

Function name      **\_\_STATIC\_INLINE void LL\_FLASH\_EnableHalfCycleAccess  
(void )**

Function description      Enable Flash Half Cycle Access.

Return values      •    **None**

Reference Manual to  
LL API cross  
reference:

### **LL\_FLASH\_DisableHalfCycleAccess**

Function name      **\_\_STATIC\_INLINE void LL\_FLASH\_DisableHalfCycleAccess  
(void )**

Function description      Disable Flash Half Cycle Access.

Return values      •    **None**

- Reference Manual to • FLASH\_ACR HLF CYA LL\_FLASH\_DisableHalfCycleAccess  
LL API cross reference:

### **LL\_FLASH\_IsHalfCycleAccessEnabled**

|                                                   |                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_FLASH_IsHalfCycleAccessEnabled (void )</code>                         |
| Function description                              | Check if Flash Half Cycle Access is enabled or not.                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• FLASH_ACR HLF CYA<br/>LL_FLASH_IsHalfCycleAccessEnabled</li> </ul> |

## **77.2 SYSTEM Firmware driver defines**

### **77.2.1 SYSTEM**

#### ***SYSCFG ADC DMA request REMAP***

|                                            |                                                    |
|--------------------------------------------|----------------------------------------------------|
| <code>LL_SYSCFG_ADC24_RMP_DMA2_CH12</code> | ADC24 DMA requests mapped on DMA2 channels 1 and 2 |
| <code>LL_SYSCFG_ADC24_RMP_DMA2_CH34</code> | ADC24 DMA requests mapped on DMA2 channels 3 and 4 |

#### ***DBGMCU APB1 GRP1 STOP IP***

|                                            |                                                        |
|--------------------------------------------|--------------------------------------------------------|
| <code>LL_DBGMCU_APB1_GRP1_TIM2_STOP</code> | TIM2 counter stopped when core is halted               |
| <code>LL_DBGMCU_APB1_GRP1_TIM3_STOP</code> | TIM3 counter stopped when core is halted               |
| <code>LL_DBGMCU_APB1_GRP1_TIM4_STOP</code> | TIM4 counter stopped when core is halted               |
| <code>LL_DBGMCU_APB1_GRP1_TIM6_STOP</code> | TIM6 counter stopped when core is halted               |
| <code>LL_DBGMCU_APB1_GRP1_TIM7_STOP</code> | TIM7 counter stopped when core is halted               |
| <code>LL_DBGMCU_APB1_GRP1_RTC_STOP</code>  | RTC counter stopped when core is halted                |
| <code>LL_DBGMCU_APB1_GRP1_WWDG_STOP</code> | Debug Window Watchdog stopped when Core is halted      |
| <code>LL_DBGMCU_APB1_GRP1_IWDG_STOP</code> | Debug Independent Watchdog stopped when Core is halted |
| <code>LL_DBGMCU_APB1_GRP1_I2C1_STOP</code> | I2C1 SMBUS timeout mode stopped when Core is halted    |
| <code>LL_DBGMCU_APB1_GRP1_I2C2_STOP</code> | I2C2 SMBUS timeout mode stopped when Core is halted    |
| <code>LL_DBGMCU_APB1_GRP1_CAN_STOP</code>  | CAN debug stopped when Core is halted                  |

#### ***DBGMCU APB2 GRP1 STOP IP***

|                                             |                                           |
|---------------------------------------------|-------------------------------------------|
| <code>LL_DBGMCU_APB2_GRP1_TIM1_STOP</code>  | TIM1 counter stopped when core is halted  |
| <code>LL_DBGMCU_APB2_GRP1_TIM8_STOP</code>  | TIM8 counter stopped when core is halted  |
| <code>LL_DBGMCU_APB2_GRP1_TIM15_STOP</code> | TIM15 counter stopped when core is halted |

`LL_DBGMCU_APB2_GRP1_TIM16_STOP` TIM16 counter stopped when core is halted

`LL_DBGMCU_APB2_GRP1_TIM17_STOP` TIM17 counter stopped when core is halted

#### ***SYSCFG CCM SRAM WRP***

`LL_SYSCFG_CCMSRAMWRP_PAGE0` ICODE SRAM Write protection page 0

`LL_SYSCFG_CCMSRAMWRP_PAGE1` ICODE SRAM Write protection page 1

`LL_SYSCFG_CCMSRAMWRP_PAGE2` ICODE SRAM Write protection page 2

`LL_SYSCFG_CCMSRAMWRP_PAGE3` ICODE SRAM Write protection page 3

`LL_SYSCFG_CCMSRAMWRP_PAGE4` ICODE SRAM Write protection page 4

`LL_SYSCFG_CCMSRAMWRP_PAGE5` ICODE SRAM Write protection page 5

`LL_SYSCFG_CCMSRAMWRP_PAGE6` ICODE SRAM Write protection page 6

`LL_SYSCFG_CCMSRAMWRP_PAGE7` ICODE SRAM Write protection page 7

#### ***SYSCFG DAC1/2 DMA1/2 request REMAP***

`LL_SYSCFG_DAC1_CH1_RMP_DMA2_CH3` DAC\_CH1 DMA requests mapped on DMA2 channel 3

`LL_SYSCFG_DAC1_CH1_RMP_DMA1_CH3` DAC\_CH1 DMA requests mapped on DMA1 channel 3

`LL_SYSCFG_DAC1_OUT2_RMP_DMA2_CH4` DAC1\_OUT2 DMA requests mapped on DMA2 channel 4

`LL_SYSCFG_DAC1_OUT2_RMP_DMA1_CH4` DAC1\_OUT2 DMA requests mapped on DMA1 channel 4

#### ***SYSCFG DAC1 Trigger REMAP***

`LL_SYSCFG_DAC1_TRIG1_RMP_TIM8_TRGO` No remap: DAC trigger TRIG1 is TIM8\_TRGO

`LL_SYSCFG_DAC1_TRIG1_RMP_TIM3_TRGO` DAC trigger is TIM3\_TRGO

#### ***SYSCFG EXTI LINE***

`LL_SYSCFG_EXTI_LINE0`

`LL_SYSCFG_EXTI_LINE1`

`LL_SYSCFG_EXTI_LINE2`

`LL_SYSCFG_EXTI_LINE3`

`LL_SYSCFG_EXTI_LINE4`

`LL_SYSCFG_EXTI_LINE5`

`LL_SYSCFG_EXTI_LINE6`

`LL_SYSCFG_EXTI_LINE7`

`LL_SYSCFG_EXTI_LINE8`

`LL_SYSCFG_EXTI_LINE9`

`LL_SYSCFG_EXTI_LINE10`

`LL_SYSCFG_EXTI_LINE11`

`LL_SYSCFG_EXTI_LINE12`

LL\_SYSCFG\_EXTI\_LINE13

LL\_SYSCFG\_EXTI\_LINE14

LL\_SYSCFG\_EXTI\_LINE15

#### **SYSCFG EXTI PORT**

LL\_SYSCFG\_EXTI\_PORTA EXTI PORT A

LL\_SYSCFG\_EXTI\_PORTB EXTI PORT B

LL\_SYSCFG\_EXTI\_PORTC EXTI PORT C

LL\_SYSCFG\_EXTI\_PORTD EXTI PORT D

LL\_SYSCFG\_EXTI PORTE EXTI PORT E

LL\_SYSCFG\_EXTI\_PORTF EXTI PORT F

#### **SYSCFG I2C FASTMODEPLUS**

LL\_SYSCFG\_I2C\_FASTMODEPLUS\_PB6 I2C PB6 Fast mode plus

LL\_SYSCFG\_I2C\_FASTMODEPLUS\_PB7 I2C PB7 Fast mode plus

LL\_SYSCFG\_I2C\_FASTMODEPLUS\_PB8 I2C PB8 Fast mode plus

LL\_SYSCFG\_I2C\_FASTMODEPLUS\_PB9 I2C PB9 Fast mode plus

LL\_SYSCFG\_I2C\_FASTMODEPLUS\_I2C1 I2C1 Fast mode plus

LL\_SYSCFG\_I2C\_FASTMODEPLUS\_I2C2 I2C2 Fast mode plus

#### **FLASH LATENCY**

LL\_FLASH\_LATENCY\_0 FLASH Zero Latency cycle

LL\_FLASH\_LATENCY\_1 FLASH One Latency cycle

LL\_FLASH\_LATENCY\_2 FLASH Two Latency cycles

#### **SYSCFG REMAP**

LL\_SYSCFG\_REMAP\_FLASH

LL\_SYSCFG\_REMAP\_SYSTEMFLASH

LL\_SYSCFG\_REMAP\_SRAM

#### **SYSCFG TIM DMA request REMAP**

LL\_SYSCFG\_TIM16\_RMP\_DMA1\_CH3 TIM16\_CH1 and TIM16\_UP DMA requests mapped on DMA1 channel 3

LL\_SYSCFG\_TIM16\_RMP\_DMA1\_CH6 TIM16\_CH1 and TIM16\_UP DMA requests mapped on DMA1 channel 6

LL\_SYSCFG\_TIM17\_RMP\_DMA1\_CH1 TIM17\_CH1 and TIM17\_UP DMA requests mapped on DMA1 channel 1

LL\_SYSCFG\_TIM17\_RMP\_DMA1\_CH7 TIM17\_CH1 and TIM17\_UP DMA requests mapped on DMA1 channel 7

LL\_SYSCFG\_TIM6\_RMP\_DMA2\_CH3 TIM6 DMA requests mapped on DMA2 channel 3

LL\_SYSCFG\_TIM6\_RMP\_DMA1\_CH3 TIM6 DMA requests mapped on DMA1 channel 3

|                             |                                            |
|-----------------------------|--------------------------------------------|
| LL_SYSCFG_TIM7_RMP_DMA2_CH4 | TIM7 DMA requests mapped on DMA2 channel 4 |
| LL_SYSCFG_TIM7_RMP_DMA1_CH4 | TIM7 DMA requests mapped on DMA1 channel 4 |

**SYSCFG TIM REMAP**

|                                          |                                                                             |
|------------------------------------------|-----------------------------------------------------------------------------|
| LL_SYSCFG_TIM1_ITR3_RMP_TIM4_TRGO        | TIM1_ITR3 = TIM4_TRGO                                                       |
| LL_SYSCFG_TIM1_ITR3_RMP_TIM17_OC         | TIM1_ITR3 = TIM17_OC                                                        |
| LL_SYSCFG_TIM15_ENCODEMODE_NOREDIRECTION | No redirection                                                              |
| LL_SYSCFG_TIM15_ENCODEMODE_TIM2          | TIM2 IC1 and TIM2 IC2 are connected to TIM15 IC1 and TIM15 IC2 respectively |
| LL_SYSCFG_TIM15_ENCODEMODE_TIM3          | TIM3 IC1 and TIM3 IC2 are connected to TIM15 IC1 and TIM15 IC2 respectively |
| LL_SYSCFG_TIM15_ENCODEMODE_TIM4          | TIM4 IC1 and TIM4 IC2 are connected to TIM15 IC1 and TIM15 IC2 respectively |

**SYSCFG TIMER BREAK**

|                                |                                                                                                                          |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| LL_SYSCFG_TIMBREAK_PVD         | Enables and locks the PVD connection with TIMx Break Input and also the PVDE and PLS bits of the Power Control Interface |
| LL_SYSCFG_TIMBREAK_SRAM_PARITY | Enables and locks the SRAM_PARITY error signal with Break Input of TIMx                                                  |
| LL_SYSCFG_TIMBREAK_LOCKUP      | Enables and locks the LOCKUP (Hardfault) output of CortexM0 with Break Input of TIMx                                     |

**DBGMCU TRACE Pin Assignment**

|                             |                                                                      |
|-----------------------------|----------------------------------------------------------------------|
| LL_DBGMCU_TRACE_NONE        | TRACE pins not assigned (default state)                              |
| LL_DBGMCU_TRACE_ASYNCNCH    | TRACE pin assignment for Asynchronous Mode                           |
| LL_DBGMCU_TRACE_SYNCH_SIZE1 | TRACE pin assignment for Synchronous Mode with a TRACEDATA size of 1 |
| LL_DBGMCU_TRACE_SYNCH_SIZE2 | TRACE pin assignment for Synchronous Mode with a TRACEDATA size of 2 |
| LL_DBGMCU_TRACE_SYNCH_SIZE4 | TRACE pin assignment for Synchronous Mode with a TRACEDATA size of 4 |

## 78 LL TIM Generic Driver

### 78.1 TIM Firmware driver registers structures

#### 78.1.1 LL\_TIM\_InitTypeDef

##### Data Fields

- *uint16\_t Prescaler*
- *uint32\_t CounterMode*
- *uint32\_t Autoreload*
- *uint32\_t ClockDivision*
- *uint8\_t RepetitionCounter*

##### Field Documentation

- ***uint16\_t LL\_TIM\_InitTypeDef::Prescaler***

Specifies the prescaler value used to divide the TIM clock. This parameter can be a number between Min\_Data=0x0000 and Max\_Data=0xFFFF. This feature can be modified afterwards using unitary function **LL\_TIM\_SetPrescaler()**.

- ***uint32\_t LL\_TIM\_InitTypeDef::CounterMode***

Specifies the counter mode. This parameter can be a value of **TIM\_LL\_EC\_COUNTERMODE**. This feature can be modified afterwards using unitary function **LL\_TIM\_SetCounterMode()**.

- ***uint32\_t LL\_TIM\_InitTypeDef::Autoreload***

Specifies the auto reload value to be loaded into the active Auto-Reload Register at the next update event. This parameter must be a number between Min\_Data=0x0000 and Max\_Data=0xFFFF. Some timer instances may support 32 bits counters. In that case this parameter must be a number between 0x0000 and 0xFFFFFFFF. This feature can be modified afterwards using unitary function **LL\_TIM\_SetAutoReload()**.

- ***uint32\_t LL\_TIM\_InitTypeDef::ClockDivision***

Specifies the clock division. This parameter can be a value of **TIM\_LL\_EC\_CLOCKDIVISION**. This feature can be modified afterwards using unitary function **LL\_TIM\_SetClockDivision()**.

- ***uint8\_t LL\_TIM\_InitTypeDef::RepetitionCounter***

Specifies the repetition counter value. Each time the RCR downcounter reaches zero, an update event is generated and counting restarts from the RCR value (N). This means in PWM mode that (N+1) corresponds to the number of PWM periods in edge-aligned mode the number of half PWM period in center-aligned mode. This parameter must be a number between 0x00 and 0xFF. This feature can be modified afterwards using unitary function **LL\_TIM\_SetRepetitionCounter()**.

#### 78.1.2 LL\_TIM\_OC\_InitTypeDef

##### Data Fields

- *uint32\_t OCMode*
- *uint32\_t OCState*
- *uint32\_t OCNState*
- *uint32\_t CompareValue*
- *uint32\_t OCIdleState*
- *uint32\_t OCNPolarity*
- *uint32\_t OCIdleState*
- *uint32\_t OCNPolarity*

### Field Documentation

- ***uint32\_t LL\_TIM\_OC\_InitTypeDef::OCMode***  
Specifies the output mode. This parameter can be a value of ***TIM\_LL\_EC\_OCMODE***. This feature can be modified afterwards using unitary function ***LL\_TIM\_OC\_SetMode()***.
- ***uint32\_t LL\_TIM\_OC\_InitTypeDef::OCState***  
Specifies the TIM Output Compare state. This parameter can be a value of ***TIM\_LL\_EC\_OCSTATE***. This feature can be modified afterwards using unitary functions ***LL\_TIM\_CC\_EnableChannel()*** or ***LL\_TIM\_CC\_DisableChannel()***.
- ***uint32\_t LL\_TIM\_OC\_InitTypeDef::OCNState***  
Specifies the TIM complementary Output Compare state. This parameter can be a value of ***TIM\_LL\_EC\_OCSTATE***. This feature can be modified afterwards using unitary functions ***LL\_TIM\_CC\_EnableChannel()*** or ***LL\_TIM\_CC\_DisableChannel()***.
- ***uint32\_t LL\_TIM\_OC\_InitTypeDef::CompareValue***  
Specifies the Compare value to be loaded into the Capture Compare Register. This parameter can be a number between Min\_Data=0x0000 and Max\_Data=0xFFFF. This feature can be modified afterwards using unitary function ***LL\_TIM\_OC\_SetCompareCHx*** ( $x=1..6$ ).
- ***uint32\_t LL\_TIM\_OC\_InitTypeDef::OCPolarity***  
Specifies the output polarity. This parameter can be a value of ***TIM\_LL\_EC\_OCPOLARITY***. This feature can be modified afterwards using unitary function ***LL\_TIM\_OC\_SetPolarity()***.
- ***uint32\_t LL\_TIM\_OC\_InitTypeDef::OCNPolarity***  
Specifies the complementary output polarity. This parameter can be a value of ***TIM\_LL\_EC\_OCPOLARITY***. This feature can be modified afterwards using unitary function ***LL\_TIM\_OC\_SetPolarity()***.
- ***uint32\_t LL\_TIM\_OC\_InitTypeDef::OCIdleState***  
Specifies the TIM Output Compare pin state during Idle state. This parameter can be a value of ***TIM\_LL\_EC\_OCIDLESTATE***. This feature can be modified afterwards using unitary function ***LL\_TIM\_OC\_SetIdleState()***.
- ***uint32\_t LL\_TIM\_OC\_InitTypeDef::OCNIdleState***  
Specifies the TIM Output Compare pin state during Idle state. This parameter can be a value of ***TIM\_LL\_EC\_OCIDLESTATE***. This feature can be modified afterwards using unitary function ***LL\_TIM\_OC\_SetIdleState()***.

### 78.1.3 LL\_TIM\_IC\_InitTypeDef

#### Data Fields

- ***uint32\_t IC\_Polarity***
- ***uint32\_t IC\_ActiveInput***
- ***uint32\_t IC\_Prescaler***
- ***uint32\_t IC\_Filter***

#### Field Documentation

- ***uint32\_t LL\_TIM\_IC\_InitTypeDef::IC\_Polarity***  
Specifies the active edge of the input signal. This parameter can be a value of ***TIM\_LL\_EC\_IC\_POLARITY***. This feature can be modified afterwards using unitary function ***LL\_TIM\_IC\_SetPolarity()***.
- ***uint32\_t LL\_TIM\_IC\_InitTypeDef::IC\_ActiveInput***  
Specifies the input. This parameter can be a value of ***TIM\_LL\_EC\_ACTIVEINPUT***. This feature can be modified afterwards using unitary function ***LL\_TIM\_IC\_SetActiveInput()***.
- ***uint32\_t LL\_TIM\_IC\_InitTypeDef::IC\_Prescaler***  
Specifies the Input Capture Prescaler. This parameter can be a value of

- TIM\_LL\_EC\_ICPSC***. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetPrescaler()**.
- ***uint32\_t LL\_TIM\_IC\_InitTypeDef::ICFilter***  
Specifies the input capture filter. This parameter can be a value of ***TIM\_LL\_EC\_IC\_FILTER***. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetFilter()**.

#### 78.1.4 **LL\_TIM\_ENCODER\_InitTypeDef**

##### Data Fields

- ***uint32\_t EncoderMode***
- ***uint32\_t IC1Polarity***
- ***uint32\_t IC1ActiveInput***
- ***uint32\_t IC1Prescaler***
- ***uint32\_t IC1Filter***
- ***uint32\_t IC2Polarity***
- ***uint32\_t IC2ActiveInput***
- ***uint32\_t IC2Prescaler***
- ***uint32\_t IC2Filter***

##### Field Documentation

- ***uint32\_t LL\_TIM\_ENCODER\_InitTypeDef::EncoderMode***  
Specifies the encoder resolution (x2 or x4). This parameter can be a value of ***TIM\_LL\_EC\_ENCODERMODE***. This feature can be modified afterwards using unitary function **LL\_TIM\_SetEncoderMode()**.
- ***uint32\_t LL\_TIM\_ENCODER\_InitTypeDef::IC1Polarity***  
Specifies the active edge of TI1 input. This parameter can be a value of ***TIM\_LL\_EC\_IC\_POLARITY***. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetPolarity()**.
- ***uint32\_t LL\_TIM\_ENCODER\_InitTypeDef::IC1ActiveInput***  
Specifies the TI1 input source. This parameter can be a value of ***TIM\_LL\_EC\_ACTIVEINPUT***. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetActiveInput()**.
- ***uint32\_t LL\_TIM\_ENCODER\_InitTypeDef::IC1Prescaler***  
Specifies the TI1 input prescaler value. This parameter can be a value of ***TIM\_LL\_EC\_ICPSC***. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetPrescaler()**.
- ***uint32\_t LL\_TIM\_ENCODER\_InitTypeDef::IC1Filter***  
Specifies the TI1 input filter. This parameter can be a value of ***TIM\_LL\_EC\_IC\_FILTER***. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetFilter()**.
- ***uint32\_t LL\_TIM\_ENCODER\_InitTypeDef::IC2Polarity***  
Specifies the active edge of TI2 input. This parameter can be a value of ***TIM\_LL\_EC\_IC\_POLARITY***. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetPolarity()**.
- ***uint32\_t LL\_TIM\_ENCODER\_InitTypeDef::IC2ActiveInput***  
Specifies the TI2 input source. This parameter can be a value of ***TIM\_LL\_EC\_ACTIVEINPUT***. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetActiveInput()**.
- ***uint32\_t LL\_TIM\_ENCODER\_InitTypeDef::IC2Prescaler***  
Specifies the TI2 input prescaler value. This parameter can be a value of ***TIM\_LL\_EC\_ICPSC***. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetPrescaler()**.

- ***uint32\_t LL\_TIM\_ENCODER\_InitTypeDef::IC2Filter***  
Specifies the TI2 input filter. This parameter can be a value of **TIM\_LL\_EC\_IC\_FILTER**. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetFilter()**.

### 78.1.5 LL\_TIM\_HALLSENSOR\_InitTypeDef

#### Data Fields

- ***uint32\_t IC1Polarity***
- ***uint32\_t IC1Prescaler***
- ***uint32\_t IC1Filter***
- ***uint32\_t CommutationDelay***

#### Field Documentation

- ***uint32\_t LL\_TIM\_HALLSENSOR\_InitTypeDef::IC1Polarity***  
Specifies the active edge of TI1 input. This parameter can be a value of **TIM\_LL\_EC\_IC\_POLARITY**. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetPolarity()**.
- ***uint32\_t LL\_TIM\_HALLSENSOR\_InitTypeDef::IC1Prescaler***  
Specifies the TI1 input prescaler value. Prescaler must be set to get a maximum counter period longer than the time interval between 2 consecutive changes on the Hall inputs. This parameter can be a value of **TIM\_LL\_EC\_ICPSC**. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetPrescaler()**.
- ***uint32\_t LL\_TIM\_HALLSENSOR\_InitTypeDef::IC1Filter***  
Specifies the TI1 input filter. This parameter can be a value of **TIM\_LL\_EC\_IC\_FILTER**. This feature can be modified afterwards using unitary function **LL\_TIM\_IC\_SetFilter()**.
- ***uint32\_t LL\_TIM\_HALLSENSOR\_InitTypeDef::CommutationDelay***  
Specifies the compare value to be loaded into the Capture Compare Register. A positive pulse (TRGO event) is generated with a programmable delay every time a change occurs on the Hall inputs. This parameter can be a number between Min\_Data = 0x0000 and Max\_Data = 0xFFFF. This feature can be modified afterwards using unitary function **LL\_TIM\_OC\_SetCompareCH2()**.

### 78.1.6 LL\_TIM\_BDTR\_InitTypeDef

#### Data Fields

- ***uint32\_t OSSRState***
- ***uint32\_t OSSISState***
- ***uint32\_t LockLevel***
- ***uint8\_t DeadTime***
- ***uint16\_t BreakState***
- ***uint32\_t BreakPolarity***
- ***uint32\_t BreakFilter***
- ***uint32\_t Break2State***
- ***uint32\_t Break2Polarity***
- ***uint32\_t Break2Filter***
- ***uint32\_t AutomaticOutput***

#### Field Documentation

- ***uint32\_t LL\_TIM\_BDTR\_InitTypeDef::OSSRState***  
Specifies the Off-State selection used in Run mode. This parameter can be a value of **TIM\_LL\_EC\_OSSR**. This feature can be modified afterwards using unitary function **LL\_TIM\_SetOffStates()**.

- Note:**This bit-field cannot be modified as long as LOCK level 2 has been programmed.
- ***uint32\_t LL\_TIM\_BDTR\_InitTypeDef::OSSIState***  
Specifies the Off-State used in Idle state. This parameter can be a value of ***TIM\_LL\_EC\_OSSI***This feature can be modified afterwards using unitary function ***LL\_TIM\_SetOffStates()***  
**Note:**This bit-field cannot be modified as long as LOCK level 2 has been programmed.
  - ***uint32\_t LL\_TIM\_BDTR\_InitTypeDef::LockLevel***  
Specifies the LOCK level parameters. This parameter can be a value of ***TIM\_LL\_EC\_LOCKLEVEL***  
**Note:**The LOCK bits can be written only once after the reset. Once the TIMx\_BDTR register has been written, their content is frozen until the next reset.
  - ***uint8\_t LL\_TIM\_BDTR\_InitTypeDef::DeadTime***  
Specifies the delay time between the switching-off and the switching-on of the outputs. This parameter can be a number between Min\_Data = 0x00 and Max\_Data = 0xFF.This feature can be modified afterwards using unitary function ***LL\_TIM\_OC\_SetDeadTime()***  
**Note:**This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed.
  - ***uint16\_t LL\_TIM\_BDTR\_InitTypeDef::BreakState***  
Specifies whether the TIM Break input is enabled or not. This parameter can be a value of ***TIM\_LL\_EC\_BREAK\_ENABLE***This feature can be modified afterwards using unitary functions ***LL\_TIM\_EnableBRK()*** or ***LL\_TIM\_DisableBRK()***  
**Note:**This bit-field can not be modified as long as LOCK level 1 has been programmed.
  - ***uint32\_t LL\_TIM\_BDTR\_InitTypeDef::BreakPolarity***  
Specifies the TIM Break Input pin polarity. This parameter can be a value of ***TIM\_LL\_EC\_BREAK\_POLARITY***This feature can be modified afterwards using unitary function ***LL\_TIM\_ConfigBRK()***  
**Note:**This bit-field can not be modified as long as LOCK level 1 has been programmed.
  - ***uint32\_t LL\_TIM\_BDTR\_InitTypeDef::BreakFilter***  
Specifies the TIM Break Filter. This parameter can be a value of ***TIM\_LL\_EC\_BREAK\_FILTER***This feature can be modified afterwards using unitary function ***LL\_TIM\_ConfigBRK()***  
**Note:**This bit-field can not be modified as long as LOCK level 1 has been programmed.
  - ***uint32\_t LL\_TIM\_BDTR\_InitTypeDef::Break2State***  
Specifies whether the TIM Break2 input is enabled or not. This parameter can be a value of ***TIM\_LL\_EC\_BREAK2\_ENABLE***This feature can be modified afterwards using unitary functions ***LL\_TIM\_EnableBRK2()*** or ***LL\_TIM\_DisableBRK2()***  
**Note:**This bit-field can not be modified as long as LOCK level 1 has been programmed.
  - ***uint32\_t LL\_TIM\_BDTR\_InitTypeDef::Break2Polarity***  
Specifies the TIM Break2 Input pin polarity. This parameter can be a value of ***TIM\_LL\_EC\_BREAK2\_POLARITY***This feature can be modified afterwards using unitary function ***LL\_TIM\_ConfigBRK2()***  
**Note:**This bit-field can not be modified as long as LOCK level 1 has been programmed.
  - ***uint32\_t LL\_TIM\_BDTR\_InitTypeDef::Break2Filter***  
Specifies the TIM Break2 Filter. This parameter can be a value of ***TIM\_LL\_EC\_BREAK2\_FILTER***This feature can be modified afterwards using unitary function ***LL\_TIM\_ConfigBRK2()***

**Note:**This bit-field can not be modified as long as LOCK level 1 has been programmed.

- **`uint32_t LL_TIM_BDTR_InitTypeDef::AutomaticOutput`**

Specifies whether the TIM Automatic Output feature is enabled or not. This parameter can be a value of `TIM_LL_EC_AUTOMATICOUTPUT_ENABLE`. This feature can be modified afterwards using unitary functions `LL_TIM_EnableAutomaticOutput()` or `LL_TIM_DisableAutomaticOutput()`.

**Note:**This bit-field can not be modified as long as LOCK level 1 has been programmed.

## 78.2 TIM Firmware driver API description

### 78.2.1 Detailed description of functions

#### LL\_TIM\_EnableCounter

Function name      **`_STATIC_INLINE void LL_TIM_EnableCounter (TIM_TypeDef * TIMx)`**

Function description      Enable timer counter.

Parameters      • **TIMx:** Timer instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR1 CEN LL\_TIM\_EnableCounter

#### LL\_TIM\_DisableCounter

Function name      **`_STATIC_INLINE void LL_TIM_DisableCounter (TIM_TypeDef * TIMx)`**

Function description      Disable timer counter.

Parameters      • **TIMx:** Timer instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR1 CEN LL\_TIM\_DisableCounter

#### LL\_TIM\_IsEnabledCounter

Function name      **`_STATIC_INLINE uint32_t LL_TIM_IsEnabledCounter (TIM_TypeDef * TIMx)`**

Function description      Indicates whether the timer counter is enabled.

Parameters      • **TIMx:** Timer instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
• CR1 CEN LL\_TIM\_IsEnabledCounter

**LL\_TIM\_EnableUpdateEvent**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_EnableUpdateEvent<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Enable update event generation.                                                           |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR1 UDIS LL_TIM_EnableUpdateEvent</li></ul>       |

**LL\_TIM\_DisableUpdateEvent**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_DisableUpdateEvent<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Disable update event generation.                                                           |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>              |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR1 UDIS LL_TIM_DisableUpdateEvent</li></ul>       |

**LL\_TIM\_IsEnabledUpdateEvent**

|                                                   |                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_TIM_IsEnabledUpdateEvent<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Indicates whether update event generation is enabled.                                            |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>                    |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR1 UDIS LL_TIM_IsEnabledUpdateEvent</li></ul>           |

**LL\_TIM\_SetUpdateSource**

|                      |                                                                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_TIM_SetUpdateSource<br/>(TIM_TypeDef * TIMx, uint32_t UpdateSource)</code></b>                                                                                                                                                                     |
| Function description | Set update event source.                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li><li>• <b>UpdateSource:</b> This parameter can be one of the following values:<ul style="list-style-type: none"><li>– LL_TIM_UPDATESOURCE_REGULAR</li><li>– LL_TIM_UPDATESOURCE_COUNTER</li></ul></li></ul> |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"><li>• Update event source set to LL_TIM_UPDATESOURCE_REGULAR: any of the following</li></ul>                                                                                                                                                     |

events generate an update interrupt or DMA request if enabled: Counter overflow/underflowSetting the UG bitUpdate generation through the slave mode controller

- Update event source set to LL\_TIM\_UPDATESOURCE\_COUNTER: only counter overflow/underflow generates an update interrupt or DMA request if enabled.

Reference Manual to  
LL API cross  
reference:

- CR1 URS LL\_TIM\_SetUpdateSource

### **LL\_TIM\_GetUpdateSource**

Function name **`_STATIC_INLINE uint32_t LL_TIM_GetUpdateSource(  
TIM_TypeDef * TIMx)`**

Function description Get actual event update source.

Parameters • **TIMx:** Timer instance

Return values • **Returned:** value can be one of the following values:  
– LL\_TIM\_UPDATESOURCE\_REGULAR  
– LL\_TIM\_UPDATESOURCE\_COUNTER

Reference Manual to  
LL API cross  
reference:

- CR1 URS LL\_TIM\_GetUpdateSource

### **LL\_TIM\_SetOnePulseMode**

Function name **`_STATIC_INLINE void LL_TIM_SetOnePulseMode(  
TIM_TypeDef * TIMx, uint32_t OnePulseMode)`**

Function description Set one pulse mode (one shot v.s.

Parameters • **TIMx:** Timer instance  
• **OnePulseMode:** This parameter can be one of the following values:  
– LL\_TIM\_ONEPULSEMODE\_SINGLE  
– LL\_TIM\_ONEPULSEMODE\_REPETITIVE

Return values • **None**

Reference Manual to  
LL API cross  
reference:

- CR1 OPM LL\_TIM\_SetOnePulseMode

### **LL\_TIM\_GetOnePulseMode**

Function name **`_STATIC_INLINE uint32_t LL_TIM_GetOnePulseMode(  
TIM_TypeDef * TIMx)`**

Function description Get actual one pulse mode.

Parameters • **TIMx:** Timer instance

Return values • **Returned:** value can be one of the following values:  
– LL\_TIM\_ONEPULSEMODE\_SINGLE

- LL\_TIM\_ONEPULSEMODE\_REPEATITIVE

Reference Manual to  
LL API cross  
reference:

- CR1 OPM LL\_TIM\_SetOnePulseMode

### **LL\_TIM\_SetCounterMode**

Function name      **`_STATIC_INLINE void LL_TIM_SetCounterMode(  
(TIM_TypeDef * TIMx, uint32_t CounterMode)`**

Function description      Set the timer counter counting mode.

- Parameters
- **TIMx:** Timer instance
  - **CounterMode:** This parameter can be one of the following values:
    - LL\_TIM\_COUNTERMODE\_UP
    - LL\_TIM\_COUNTERMODE\_DOWN
    - LL\_TIM\_COUNTERMODE\_CENTER\_UP
    - LL\_TIM\_COUNTERMODE\_CENTER\_DOWN
    - LL\_TIM\_COUNTERMODE\_CENTER\_UP\_DOWN

- Return values
- **None**

Notes

- Macro  
`IS_TIM_COUNTER_MODE_SELECT_INSTANCE(TIMx)` can be used to check whether or not the counter mode selection feature is supported by a timer instance.

Reference Manual to  
LL API cross  
reference:

- CR1 DIR LL\_TIM\_SetCounterMode
- CR1 CMS LL\_TIM\_SetCounterMode

### **LL\_TIM\_GetCounterMode**

Function name      **`_STATIC_INLINE uint32_t LL_TIM_GetCounterMode(  
(TIM_TypeDef * TIMx)`**

Function description      Get actual counter mode.

Parameters

- **TIMx:** Timer instance

- Return values
- **Returned:** value can be one of the following values:
    - LL\_TIM\_COUNTERMODE\_UP
    - LL\_TIM\_COUNTERMODE\_DOWN
    - LL\_TIM\_COUNTERMODE\_CENTER\_UP
    - LL\_TIM\_COUNTERMODE\_CENTER\_DOWN
    - LL\_TIM\_COUNTERMODE\_CENTER\_UP\_DOWN

Notes

- Macro  
`IS_TIM_COUNTER_MODE_SELECT_INSTANCE(TIMx)` can be used to check whether or not the counter mode selection feature is supported by a timer instance.

Reference Manual to  
LL API cross  
reference:

- CR1 DIR LL\_TIM\_GetCounterMode
- CR1 CMS LL\_TIM\_GetCounterMode

**LL\_TIM\_EnableARRPreload**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableARRPreload(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Enable auto-reload (ARR) preload.                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 ARPE LL_TIM_EnableARRPreload</li> </ul>   |

**LL\_TIM\_DisableARRPreload**

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableARRPreload(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Disable auto-reload (ARR) preload.                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 ARPE LL_TIM_DisableARRPreload</li> </ul>   |

**LL\_TIM\_IsEnabledARRPreload**

|                                                   |                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledARRPreload(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether auto-reload (ARR) preload is enabled.                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 ARPE LL_TIM_IsEnabledARRPreload</li> </ul>       |

**LL\_TIM\_SetClockDivision**

|                      |                                                                                                                                                                                                                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_SetClockDivision(<br/>    TIM_TypeDef * TIMx, uint32_t ClockDivision)</code>                                                                                                                                                                                                               |
| Function description | Set the division ratio between the timer clock and the sampling clock used by the dead-time generators (when supported) and the digital filters.                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>ClockDivision:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CLOCKDIVISION_DIV1</li> <li>– LL_TIM_CLOCKDIVISION_DIV2</li> <li>– LL_TIM_CLOCKDIVISION_DIV4</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_TIM_CLOCK_DIVISION_INSTANCE(TIMx) can be used to check whether or not the clock division feature is supported by the timer instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 CKD LL_TIM_SetClockDivision</li> </ul>                                                                                                               |

### LL\_TIM\_GetClockDivision

|                                                   |                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_GetClockDivision(<br/>    TIM_TypeDef * TIMx)</code>                                                                                                                                                                                      |
| Function description                              | Get the actual division ratio between the timer clock and the sampling clock used by the dead-time generators (when supported) and the digital filters.                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_TIM_CLOCKDIVISION_DIV1</li> <li>- LL_TIM_CLOCKDIVISION_DIV2</li> <li>- LL_TIM_CLOCKDIVISION_DIV4</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_TIM_CLOCK_DIVISION_INSTANCE(TIMx) can be used to check whether or not the clock division feature is supported by the timer instance.</li> </ul>                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 CKD LL_TIM_GetClockDivision</li> </ul>                                                                                                                                                                                               |

### LL\_TIM\_SetCounter

|                                                   |                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_SetCounter (TIM_TypeDef *<br/>    TIMx, uint32_t Counter)</code>                                                                          |
| Function description                              | Set the counter value.                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer instance</li> <li><b>Counter:</b> Counter value (between Min_Data=0 and Max_Data=0xFFFF or 0xFFFFFFFF)</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> </ul>  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CNT CNT LL_TIM_SetCounter</li> </ul>                                                                                                 |

### LL\_TIM\_GetCounter

|               |                                                                                      |
|---------------|--------------------------------------------------------------------------------------|
| Function name | <code>__STATIC_INLINE uint32_t LL_TIM_GetCounter (TIM_TypeDef<br/>    * TIMx)</code> |
|---------------|--------------------------------------------------------------------------------------|

---

|                                             |                                                                                                                                                                            |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function description                        | Get the counter value.                                                                                                                                                     |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer instance</li> </ul>                                                                                              |
| Return values                               | <ul style="list-style-type: none"> <li><b>Counter:</b> value (between Min_Data=0 and Max_Data=0xFFFF or 0xFFFFFFFF)</li> </ul>                                             |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CNT CNT LL_TIM_GetCounter</li> </ul>                                                                                                |

### LL\_TIM\_GetDirection

|                                             |                                                                                                                                                                                                                                                |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE uint32_t LL_TIM_GetDirection (TIM_TypeDef * TIMx)</code></b>                                                                                                                                                          |
| Function description                        | Get the current direction of the counter.                                                                                                                                                                                                      |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                  |
| Return values                               | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_TIM_COUNTERDIRECTION_UP</li> <li>– LL_TIM_COUNTERDIRECTION_DOWN</li> </ul> </li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 DIR LL_TIM_GetDirection</li> </ul>                                                                                                                                                                  |

### LL\_TIM\_SetPrescaler

|                                             |                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE void LL_TIM_SetPrescaler (TIM_TypeDef * TIMx, uint32_t Prescaler)</code></b>                                                                                                                                                                                                                                                                               |
| Function description                        | Set the prescaler value.                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer instance</li> <li><b>Prescaler:</b> between Min_Data=0 and Max_Data=65535</li> </ul>                                                                                                                                                                                                                                      |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                       |
| Notes                                       | <ul style="list-style-type: none"> <li>The counter clock frequency CK_CNT is equal to fCK_PSC / (PSC[15:0] + 1).</li> <li>The prescaler can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.</li> <li>Helper macro __LL_TIM_CALC_PSC can be used to calculate the Prescaler parameter</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>PSC PSC LL_TIM_SetPrescaler</li> </ul>                                                                                                                                                                                                                                                                                                       |

**LL\_TIM\_GetPrescaler**

Function name      **STATIC\_INLINE uint32\_t LL\_TIM\_GetPrescaler (TIM\_TypeDef \* TIMx)**

Function description      Get the prescaler value.

Parameters      • **TIMx:** Timer instance

Return values      • **Prescaler:** value between Min\_Data=0 and Max\_Data=65535

Reference Manual to  
LL API cross  
reference:  
• PSC PSC LL\_TIM\_GetPrescaler

**LL\_TIM\_SetAutoReload**

Function name      **STATIC\_INLINE void LL\_TIM\_SetAutoReload (TIM\_TypeDef \* TIMx, uint32\_t AutoReload)**

Function description      Set the auto-reload value.

Parameters      • **TIMx:** Timer instance  
• **AutoReload:** between Min\_Data=0 and Max\_Data=65535

Return values      • **None**

Notes      • The counter is blocked while the auto-reload value is null.  
• Macro IS\_TIM\_32B\_COUNTER\_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.  
• Helper macro \_\_LL\_TIM\_CALC\_ARR can be used to calculate the AutoReload parameter

Reference Manual to  
LL API cross  
reference:  
• ARR ARR LL\_TIM\_SetAutoReload

**LL\_TIM\_GetAutoReload**

Function name      **STATIC\_INLINE uint32\_t LL\_TIM\_GetAutoReload (TIM\_TypeDef \* TIMx)**

Function description      Get the auto-reload value.

Parameters      • **TIMx:** Timer instance

Return values      • **Auto-reload:** value

Notes      • Macro IS\_TIM\_32B\_COUNTER\_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.

Reference Manual to  
LL API cross  
reference:  
• ARR ARR LL\_TIM\_GetAutoReload

**LL\_TIM\_SetRepetitionCounter**

|                                                   |                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_SetRepetitionCounter(<br/>  TIM_TypeDef * TIMx, uint32_t RepetitionCounter)</code>                                                                                                                                                                                                |
| Function description                              | Set the repetition counter value.                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>RepetitionCounter:</b> between Min_Data=0 and Max_Data=255</li> </ul>                                                                                                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• For advanced timer instances RepetitionCounter can be up to 65535 except for STM32F373xC and STM32F378xx devices.</li> <li>• Macro IS_TIM_REPETITION_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a repetition counter.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• RCR REP LL_TIM_SetRepetitionCounter</li> </ul>                                                                                                                                                                                                                             |

**LL\_TIM\_GetRepetitionCounter**

|                                                   |                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_GetRepetitionCounter(<br/>  TIM_TypeDef * TIMx)</code>                                                                                           |
| Function description                              | Get the repetition counter value.                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Repetition:</b> counter value</li> </ul>                                                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_REPETITION_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a repetition counter.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• RCR REP LL_TIM_GetRepetitionCounter</li> </ul>                                                                                                |

**LL\_TIM\_EnableUIFRemap**

|                                                   |                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableUIFRemap(<br/>  TIM_TypeDef * TIMx)</code>                                                                                                   |
| Function description                              | Force a continuous copy of the update interrupt flag (UIF) into the timer counter register (bit 31).                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• This allows both the counter value and a potential roll-over condition signalled by the UIFCPY flag to be read in an atomic way.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 UIFREMAP LL_TIM_EnableUIFRemap</li> </ul>                                                                                               |

**LL\_TIM\_DisableUIFRemap**

Function name      **\_\_STATIC\_INLINE void LL\_TIM\_DisableUIFRemap(  
TIM\_TypeDef \* TIMx)**

Function description      Disable update interrupt flag (UIF) remapping.

Parameters      • **TIMx:** Timer instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR1 UIFREMAP LL\_TIM\_DisableUIFRemap

**LL\_TIM\_CC\_EnablePreload**

Function name      **\_\_STATIC\_INLINE void LL\_TIM\_CC\_EnablePreload(  
TIM\_TypeDef \* TIMx)**

Function description      Enable the capture/compare control bits (CCxE, CCxNE and OCxM) preload.

Parameters      • **TIMx:** Timer instance

Return values      • **None**

Notes      • CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when a commutation event (COM) occurs.  
• Only on channels that have a complementary output.  
• Macro IS\_TIM\_COMMUTATION\_EVENT\_INSTANCE(TIMx) can be used to check whether or not a timer instance is able to generate a commutation event.

Reference Manual to  
LL API cross  
reference:  
• CR2 CCPC LL\_TIM\_CC\_EnablePreload

**LL\_TIM\_CC\_DisablePreload**

Function name      **\_\_STATIC\_INLINE void LL\_TIM\_CC\_DisablePreload(  
TIM\_TypeDef \* TIMx)**

Function description      Disable the capture/compare control bits (CCxE, CCxNE and OCxM) preload.

Parameters      • **TIMx:** Timer instance

Return values      • **None**

Notes      • Macro IS\_TIM\_COMMUTATION\_EVENT\_INSTANCE(TIMx) can be used to check whether or not a timer instance is able to generate a commutation event.

Reference Manual to  
LL API cross  
reference:  
• CR2 CCPC LL\_TIM\_CC\_DisablePreload

**LL\_TIM\_CC\_SetUpdate**

|                                                   |                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_CC_SetUpdate (TIM_TypeDef * TIMx, uint32_t CCUpdateSource)</code>                                                                                                                                                                                                      |
| Function description                              | Set the updated source of the capture/compare control bits (CCxE, CCxNE and OCxM).                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>CCUpdateSource:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CCUPDATESOURCE_COMG_ONLY</li> <li>– LL_TIM_CCUPDATESOURCE_COMG_AND_TRGI</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_COMMUTATION_EVENT_INSTANCE(TIMx) can be used to check whether or not a timer instance is able to generate a commutation event.</li> </ul>                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CCUS LL_TIM_CC_SetUpdate</li> </ul>                                                                                                                                                                                                                         |

**LL\_TIM\_CC\_SetDMAReqTrigger**

|                                                   |                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_CC_SetDMAReqTrigger (TIM_TypeDef * TIMx, uint32_t DMAReqTrigger)</code>                                                                                                                                                                             |
| Function description                              | Set the trigger of the capture/compare DMA request.                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>DMAReqTrigger:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CCDMAREQUEST_CC</li> <li>– LL_TIM_CCDMAREQUEST_UPDATE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CCDS LL_TIM_CC_SetDMAReqTrigger</li> </ul>                                                                                                                                                                                               |

**LL\_TIM\_CC\_GetDMAReqTrigger**

|                                                   |                                                                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_CC_GetDMAReqTrigger (TIM_TypeDef * TIMx)</code>                                                                                                                                            |
| Function description                              | Get actual trigger of the capture/compare DMA request.                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CCDMAREQUEST_CC</li> <li>– LL_TIM_CCDMAREQUEST_UPDATE</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CCDS LL_TIM_CC_GetDMAReqTrigger</li> </ul>                                                                                                                                          |

**LL\_TIM\_CC\_SetLockLevel**

|                                                   |                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_TIM_CC_SetLockLevel<br/>(TIM_TypeDef * TIMx, uint32_t LockLevel)</b>                                                                                                                                                                                                                                      |
| Function description                              | Set the lock level to freeze the configuration of several capture/compare parameters.                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>LockLevel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_LOCKLEVEL_OFF</li> <li>– LL_TIM_LOCKLEVEL_1</li> <li>– LL_TIM_LOCKLEVEL_2</li> <li>– LL_TIM_LOCKLEVEL_3</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not the lock mechanism is supported by a timer instance.</li> </ul>                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDTR LOCK LL_TIM_CC_SetLockLevel</li> </ul>                                                                                                                                                                                                                                                |

**LL\_TIM\_CC\_EnableChannel**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_TIM_CC_EnableChannel<br/>(TIM_TypeDef * TIMx, uint32_t Channels)</b>                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                              | Enable capture/compare channels.                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channels:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH1N</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH2N</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH3N</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCER CC1E LL_TIM_CC_EnableChannel</li> <li>• CCER CC1NE LL_TIM_CC_EnableChannel</li> <li>• CCER CC2E LL_TIM_CC_EnableChannel</li> <li>• CCER CC2NE LL_TIM_CC_EnableChannel</li> <li>• CCER CC3E LL_TIM_CC_EnableChannel</li> <li>• CCER CC3NE LL_TIM_CC_EnableChannel</li> <li>• CCER CC4E LL_TIM_CC_EnableChannel</li> <li>• CCER CC5E LL_TIM_CC_EnableChannel</li> </ul>                                                                                 |

- CCER CC6E LL\_TIM\_CC\_EnableChannel

### **LL\_TIM\_CC\_DisableChannel**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_CC_DisableChannel(<br/>TIM_TypeDef * TIMx, uint32_t Channels)</code></b>                                                                                                                                                                                                                                                                                                                                                                                        |
| Function description                              | Disable capture/compare channels.                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channels:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH1N</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH2N</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH3N</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCER CC1E LL_TIM_CC_DisableChannel</li> <li>• CCER CC1NE LL_TIM_CC_DisableChannel</li> <li>• CCER CC2E LL_TIM_CC_DisableChannel</li> <li>• CCER CC2NE LL_TIM_CC_DisableChannel</li> <li>• CCER CC3E LL_TIM_CC_DisableChannel</li> <li>• CCER CC3NE LL_TIM_CC_DisableChannel</li> <li>• CCER CC4E LL_TIM_CC_DisableChannel</li> <li>• CCER CC5E LL_TIM_CC_DisableChannel</li> <li>• CCER CC6E LL_TIM_CC_DisableChannel</li> </ul>                           |

### **LL\_TIM\_CC\_IsEnabledChannel**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_TIM_CC_IsEnabledChannel(<br/>TIM_TypeDef * TIMx, uint32_t Channels)</code></b>                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description | Indicate whether channel(s) is(are) enabled.                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channels:</b> This parameter can be a combination of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH1N</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH2N</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH3N</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |

- |                                                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-----------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values<br>Notes<br>Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> <li>• CCER CC1E LL_TIM_CC_IsEnabledChannel</li> <li>• CCER CC1NE LL_TIM_CC_IsEnabledChannel</li> <li>• CCER CC2E LL_TIM_CC_IsEnabledChannel</li> <li>• CCER CC2NE LL_TIM_CC_IsEnabledChannel</li> <li>• CCER CC3E LL_TIM_CC_IsEnabledChannel</li> <li>• CCER CC3NE LL_TIM_CC_IsEnabledChannel</li> <li>• CCER CC4E LL_TIM_CC_IsEnabledChannel</li> <li>• CCER CC5E LL_TIM_CC_IsEnabledChannel</li> <li>• CCER CC6E LL_TIM_CC_IsEnabledChannel</li> </ul> |
|-----------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### LL\_TIM\_OC\_ConfigOutput

- |                                                                                                                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name<br>Function description<br>Parameters<br>Return values<br>Notes<br>Reference Manual to<br>LL API cross<br>reference: | <pre><b>__STATIC_INLINE void LL_TIM_OC_ConfigOutput</b><br/> <b>(TIM_TypeDef * TIMx, uint32_t Channel, uint32_t</b><br/> <b>Configuration)</b></pre> <p>Configure an output channel.</p> <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values:       <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH4</li> <li>- LL_TIM_CHANNEL_CH5</li> <li>- LL_TIM_CHANNEL_CH6</li> </ul> </li> <li>• <b>Configuration:</b> This parameter must be a combination of all the following values:       <ul style="list-style-type: none"> <li>- LL_TIM_OCPOLARITY_HIGH or LL_TIM_OCPOLARITY_LOW</li> <li>- LL_TIM_OCIDLESTATE_LOW or LL_TIM_OCIDLESTATE_HIGH</li> </ul> </li> <li>• <b>None</b></li> </ul> <p>CH3 CH4 CH5 and CH6 channels are not available for all F3 devices</p> <ul style="list-style-type: none"> <li>• CCMR1 CC1S LL_TIM_OC_ConfigOutput</li> <li>• CCMR1 CC2S LL_TIM_OC_ConfigOutput</li> <li>• CCMR2 CC3S LL_TIM_OC_ConfigOutput</li> <li>• CCMR2 CC4S LL_TIM_OC_ConfigOutput</li> <li>• CCMR3 CC5S LL_TIM_OC_ConfigOutput</li> <li>• CCMR3 CC6S LL_TIM_OC_ConfigOutput</li> <li>• CCER CC1P LL_TIM_OC_ConfigOutput</li> <li>• CCER CC2P LL_TIM_OC_ConfigOutput</li> <li>• CCER CC3P LL_TIM_OC_ConfigOutput</li> <li>• CCER CC4P LL_TIM_OC_ConfigOutput</li> <li>• CCER CC5P LL_TIM_OC_ConfigOutput</li> <li>• CCER CC6P LL_TIM_OC_ConfigOutput</li> <li>• CR2 OIS1 LL_TIM_OC_ConfigOutput</li> <li>• CR2 OIS2 LL_TIM_OC_ConfigOutput</li> </ul> |
|------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

- CR2 OIS3 LL\_TIM\_OC\_ConfigOutput
- CR2 OIS4 LL\_TIM\_OC\_ConfigOutput
- CR2 OIS5 LL\_TIM\_OC\_ConfigOutput
- CR2 OIS6 LL\_TIM\_OC\_ConfigOutput

### **LL\_TIM\_OC\_SetMode**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_TIM_OC_SetMode (TIM_TypeDef * TIMx,<br/>uint32_t Channel, uint32_t Mode)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                        | Define the behavior of the output reference signal OCxREF from which OCx and OCxN (when relevant) are derived.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH4</li> <li>- LL_TIM_CHANNEL_CH5</li> <li>- LL_TIM_CHANNEL_CH6</li> </ul> </li> <li>• <b>Mode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_OCMODE_FROZEN</li> <li>- LL_TIM_OCMODE_ACTIVE</li> <li>- LL_TIM_OCMODE_INACTIVE</li> <li>- LL_TIM_OCMODE_TOGGLE</li> <li>- LL_TIM_OCMODE_FORCED_INACTIVE</li> <li>- LL_TIM_OCMODE_FORCED_ACTIVE</li> <li>- LL_TIM_OCMODE_PWM1</li> <li>- LL_TIM_OCMODE_PWM2</li> <li>- LL_TIM_OCMODE_RETRIG_OPM1</li> <li>- LL_TIM_OCMODE_RETRIG_OPM2</li> <li>- LL_TIM_OCMODE_COMBINED_PWM1</li> <li>- LL_TIM_OCMODE_COMBINED_PWM2</li> <li>- LL_TIM_OCMODE_ASSYMETRIC_PWM1</li> <li>- LL_TIM_OCMODE_ASSYMETRIC_PWM2</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Notes                                       | <ul style="list-style-type: none"> <li>• The following OC modes are not available on all F3 devices :<br/> <code>LL_TIM_OCMODE_RETRIG_OPM1LL_TIM_OCMODE_RETRIG_O<br/>PM2LL_TIM_OCMODE_COMBINED_PWM1LL_TIM_OCMODE_CO<br/>MBINED_PWM2LL_TIM_OCMODE_ASSYMETRIC_PWM1LL_TIM_</code><br/> <code>OCMODE_ASSYMETRIC_PWM2</code></li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CCMR1 OC1M LL_TIM_OC_SetMode</li> <li>• CCMR1 OC2M LL_TIM_OC_SetMode</li> <li>• CCMR2 OC3M LL_TIM_OC_SetMode</li> <li>• CCMR2 OC4M LL_TIM_OC_SetMode</li> <li>• CCMR3 OC5M LL_TIM_OC_SetMode</li> <li>• CCMR3 OC6M LL_TIM_OC_SetMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

**LL\_TIM\_OC\_GetMode**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_TIM_OC_GetMode (TIM_TypeDef * TIMx,<br/>uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                        | Get the output compare mode of an output channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                      |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_OCMODE_FROZEN</li> <li>– LL_TIM_OCMODE_ACTIVE</li> <li>– LL_TIM_OCMODE_INACTIVE</li> <li>– LL_TIM_OCMODE_TOGGLE</li> <li>– LL_TIM_OCMODE_FORCED_INACTIVE</li> <li>– LL_TIM_OCMODE_FORCED_ACTIVE</li> <li>– LL_TIM_OCMODE_PWM1</li> <li>– LL_TIM_OCMODE_PWM2</li> <li>– LL_TIM_OCMODE_RETRIG_OPM1</li> <li>– LL_TIM_OCMODE_RETRIG_OPM2</li> <li>– LL_TIM_OCMODE_COMBINED_PWM1</li> <li>– LL_TIM_OCMODE_COMBINED_PWM2</li> <li>– LL_TIM_OCMODE_ASSYMETRIC_PWM1</li> <li>– LL_TIM_OCMODE_ASSYMETRIC_PWM2</li> </ul> </li> </ul> |
| Notes                                       | <ul style="list-style-type: none"> <li>• The following OC modes are not available on all F3 devices :<br/> <code>LL_TIM_OCMODE_RETRIG_OPM1</code><br/> <code>LL_TIM_OCMODE_COMBINED_PWM1</code><br/> <code>LL_TIM_OCMODE_COMBINED_PWM2</code><br/> <code>LL_TIM_OCMODE_ASSYMETRIC_PWM1</code><br/> <code>LL_TIM_OCMODE_ASSYMETRIC_PWM2</code></li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                                                                    |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CCMR1 OC1M LL_TIM_OC_SetPolarity</li> <li>• CCMR1 OC2M LL_TIM_OC_SetPolarity</li> <li>• CCMR2 OC3M LL_TIM_OC_SetPolarity</li> <li>• CCMR2 OC4M LL_TIM_OC_SetPolarity</li> <li>• CCMR3 OC5M LL_TIM_OC_SetPolarity</li> <li>• CCMR3 OC6M LL_TIM_OC_SetPolarity</li> </ul>                                                                                                                                                                                                                                                                                                                                                                 |

**LL\_TIM\_OC\_SetPolarity**

|                      |                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_OC_SetPolarity (TIM_TypeDef * TIMx, uint32_t Channel, uint32_t Polarity)</code>                                                                                                                     |
| Function description | Set the polarity of an output channel.                                                                                                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> </ul> </li> </ul> |

- LL\_TIM\_CHANNEL\_CH1N
- LL\_TIM\_CHANNEL\_CH2
- LL\_TIM\_CHANNEL\_CH2N
- LL\_TIM\_CHANNEL\_CH3
- LL\_TIM\_CHANNEL\_CH3N
- LL\_TIM\_CHANNEL\_CH4
- LL\_TIM\_CHANNEL\_CH5
- LL\_TIM\_CHANNEL\_CH6
- **Polarity:** This parameter can be one of the following values:
  - LL\_TIM\_OCPOLARITY\_HIGH
  - LL\_TIM\_OCPOLARITY\_LOW
- **None**
- CH5 and CH6 channels are not available for all F3 devices
- Reference Manual to LL API cross reference:
  - CCER CC1P LL\_TIM\_OC\_SetPolarity
  - CCER CC1NP LL\_TIM\_OC\_SetPolarity
  - CCER CC2P LL\_TIM\_OC\_SetPolarity
  - CCER CC2NP LL\_TIM\_OC\_SetPolarity
  - CCER CC3P LL\_TIM\_OC\_SetPolarity
  - CCER CC3NP LL\_TIM\_OC\_SetPolarity
  - CCER CC4P LL\_TIM\_OC\_SetPolarity
  - CCER CC5P LL\_TIM\_OC\_SetPolarity
  - CCER CC6P LL\_TIM\_OC\_SetPolarity

## LL\_TIM\_OC\_GetPolarity

- Function name `_STATIC_INLINE uint32_t LL_TIM_OC_GetPolarity(  
 TIM_TypeDef * TIMx, uint32_t Channel)`
- Function description Get the polarity of an output channel.
- Parameters
- **TIMx:** Timer instance
  - **Channel:** This parameter can be one of the following values:
    - LL\_TIM\_CHANNEL\_CH1
    - LL\_TIM\_CHANNEL\_CH1N
    - LL\_TIM\_CHANNEL\_CH2
    - LL\_TIM\_CHANNEL\_CH2N
    - LL\_TIM\_CHANNEL\_CH3
    - LL\_TIM\_CHANNEL\_CH3N
    - LL\_TIM\_CHANNEL\_CH4
    - LL\_TIM\_CHANNEL\_CH5
    - LL\_TIM\_CHANNEL\_CH6
- Return values
- **Returned:** value can be one of the following values:
    - LL\_TIM\_OCPOLARITY\_HIGH
    - LL\_TIM\_OCPOLARITY\_LOW
- Notes
- CH5 and CH6 channels are not available for all F3 devices
- Reference Manual to LL API cross reference:
  - CCER CC1P LL\_TIM\_OC\_GetPolarity
  - CCER CC1NP LL\_TIM\_OC\_GetPolarity
  - CCER CC2P LL\_TIM\_OC\_GetPolarity
  - CCER CC2NP LL\_TIM\_OC\_GetPolarity
  - CCER CC3P LL\_TIM\_OC\_GetPolarity

- CCER CC3NP LL\_TIM\_OC\_GetPolarity
- CCER CC4P LL\_TIM\_OC\_GetPolarity
- CCER CC5P LL\_TIM\_OC\_GetPolarity
- CCER CC6P LL\_TIM\_OC\_GetPolarity

### LL\_TIM\_OC\_SetIdleState

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_OC_SetIdleState(<br/>  TIM_TypeDef * TIMx, uint32_t Channel, uint32_t IdleState)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Set the IDLE state of an output channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH1N</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH2N</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH3N</li> <li>- LL_TIM_CHANNEL_CH4</li> <li>- LL_TIM_CHANNEL_CH5</li> <li>- LL_TIM_CHANNEL_CH6</li> </ul> </li> <li>• <b>IdleState:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_OCIDLESTATE_LOW</li> <li>- LL_TIM_OCIDLESTATE_HIGH</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function is significant only for the timer instances supporting the break feature. Macro <code>IS_TIM_BREAK_INSTANCE(TIMx)</code> can be used to check whether or not a timer instance provides a break input.</li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 OIS1 LL_TIM_OC_SetIdleState</li> <li>• CR2 OIS2N LL_TIM_OC_SetIdleState</li> <li>• CR2 OIS2 LL_TIM_OC_SetIdleState</li> <li>• CR2 OIS2N LL_TIM_OC_SetIdleState</li> <li>• CR2 OIS3 LL_TIM_OC_SetIdleState</li> <li>• CR2 OIS3N LL_TIM_OC_SetIdleState</li> <li>• CR2 OIS4 LL_TIM_OC_SetIdleState</li> <li>• CR2 OIS5 LL_TIM_OC_SetIdleState</li> <li>• CR2 OIS6 LL_TIM_OC_SetIdleState</li> </ul>                                                                                                                                                                                                                                          |

### LL\_TIM\_OC\_GetIdleState

|                      |                                                                                                                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_TIM_OC_GetIdleState(<br/>  TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                            |
| Function description | Get the IDLE state of an output channel.                                                                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH1N</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH2N</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH3N</li> <li>- LL_TIM_CHANNEL_CH4</li> <li>- LL_TIM_CHANNEL_CH5</li> <li>- LL_TIM_CHANNEL_CH6</li> </ul>                                                                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_OCIDLESTATE_LOW</li> <li>- LL_TIM_OCIDLESTATE_HIGH</li> </ul> </li> </ul>                                                                                                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 OIS1 LL_TIM_OC_GetIdleState</li> <li>• CR2 OIS2N LL_TIM_OC_GetIdleState</li> <li>• CR2 OIS2 LL_TIM_OC_GetIdleState</li> <li>• CR2 OIS2N LL_TIM_OC_GetIdleState</li> <li>• CR2 OIS3 LL_TIM_OC_GetIdleState</li> <li>• CR2 OIS3N LL_TIM_OC_GetIdleState</li> <li>• CR2 OIS4 LL_TIM_OC_GetIdleState</li> <li>• CR2 OIS5 LL_TIM_OC_GetIdleState</li> <li>• CR2 OIS6 LL_TIM_OC_GetIdleState</li> </ul> |

### LL\_TIM\_OC\_EnableFast

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_OC_EnableFast (TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                               |
| Function description                              | Enable fast mode for the output channel.                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH4</li> <li>- LL_TIM_CHANNEL_CH5</li> <li>- LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Acts only if the channel is configured in PWM1 or PWM2 mode.</li> <li>• OC5FE and OC6FE are not available for all F3 devices</li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 OC1FE LL_TIM_OC_EnableFast</li> <li>• CCMR1 OC2FE LL_TIM_OC_EnableFast</li> <li>• CCMR2 OC3FE LL_TIM_OC_EnableFast</li> <li>• CCMR2 OC4FE LL_TIM_OC_EnableFast</li> <li>• CCMR3 OC5FE LL_TIM_OC_EnableFast</li> <li>• CCMR3 OC6FE LL_TIM_OC_EnableFast</li> </ul>                                                                            |

**LL\_TIM\_OC\_DisableFast**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_OC_DisableFast(<br/>    TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                      |
| Function description                              | Disable fast mode for the output channel.                                                                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• OC5FE and OC6FE are not available for all F3 devices</li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 OC1FE LL_TIM_OC_DisableFast</li> <li>• CCMR1 OC2FE LL_TIM_OC_DisableFast</li> <li>• CCMR2 OC3FE LL_TIM_OC_DisableFast</li> <li>• CCMR2 OC4FE LL_TIM_OC_DisableFast</li> <li>• CCMR3 OC5FE LL_TIM_OC_DisableFast</li> <li>• CCMR3 OC6FE LL_TIM_OC_DisableFast</li> </ul>                                                                      |

**LL\_TIM\_OC\_IsEnabledFast**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_OC_IsEnabledFast(<br/>    TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                |
| Function description                              | Indicates whether fast mode is enabled for the output channel.                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• OC5FE and OC6FE are not available for all F3 devices</li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 OC1FE LL_TIM_OC_IsEnabledFast</li> <li>• CCMR1 OC2FE LL_TIM_OC_IsEnabledFast</li> <li>• CCMR2 OC3FE LL_TIM_OC_IsEnabledFast</li> <li>• CCMR2 OC4FE LL_TIM_OC_IsEnabledFast</li> <li>• CCMR3 OC5FE LL_TIM_OC_IsEnabledFast</li> <li>• CCMR3 OC6FE LL_TIM_OC_IsEnabledFast</li> </ul>                                                          |

**LL\_TIM\_OC\_EnablePreload**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_OC_EnablePreload<br/>(TIM_TypeDef * TIMx, uint32_t Channel)</code></b>                                                                                                                                                                                                                                                                                  |
| Function description                              | Enable compare register (TIMx_CCRx) preload for the output channel.                                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• OC5PE and OC6PE are not available for all F3 devices</li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 OC1PE LL_TIM_OC_EnablePreload</li> <li>• CCMR1 OC2PE LL_TIM_OC_EnablePreload</li> <li>• CCMR2 OC3PE LL_TIM_OC_EnablePreload</li> <li>• CCMR2 OC4PE LL_TIM_OC_EnablePreload</li> <li>• CCMR3 OC5PE LL_TIM_OC_EnablePreload</li> <li>• CCMR3 OC6PE LL_TIM_OC_EnablePreload</li> </ul>                                                          |

**LL\_TIM\_OC\_DisablePreload**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_OC_DisablePreload<br/>(TIM_TypeDef * TIMx, uint32_t Channel)</code></b>                                                                                                                                                                                                                                                                                 |
| Function description                              | Disable compare register (TIMx_CCRx) preload for the output channel.                                                                                                                                                                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• OC5PE and OC6PE are not available for all F3 devices</li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 OC1PE LL_TIM_OC_DisablePreload</li> <li>• CCMR1 OC2PE LL_TIM_OC_DisablePreload</li> <li>• CCMR2 OC3PE LL_TIM_OC_DisablePreload</li> <li>• CCMR2 OC4PE LL_TIM_OC_DisablePreload</li> <li>• CCMR3 OC5PE LL_TIM_OC_DisablePreload</li> <li>• CCMR3 OC6PE LL_TIM_OC_DisablePreload</li> </ul>                                                    |

**LL\_TIM\_OC\_IsEnabledPreload**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_OC_IsEnabledPreload(<br/>    TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                             |
| Function description                              | Indicates whether compare register (TIMx_CCRx) preload is enabled for the output channel.                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• OC5PE and OC6PE are not available for all F3 devices</li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 OC1PE LL_TIM_OC_IsEnabledPreload</li> <li>• CCMR1 OC2PE LL_TIM_OC_IsEnabledPreload</li> <li>• CCMR2 OC3PE LL_TIM_OC_IsEnabledPreload</li> <li>• CCMR2 OC4PE LL_TIM_OC_IsEnabledPreload</li> <li>• CCMR3 OC5PE LL_TIM_OC_IsEnabledPreload</li> <li>• CCMR3 OC6PE LL_TIM_OC_IsEnabledPreload</li> </ul>                                        |

**LL\_TIM\_OC\_EnableClear**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_OC_EnableClear(<br/>    TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                                               |
| Function description                              | Enable clearing the output channel on an external event.                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul>                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function can only be used in Output compare and PWM modes. It does not work in Forced mode.</li> <li>• Macro IS_TIM_OCXREF_CLEAR_INSTANCE(TIMx) can be used to check whether or not a timer instance can clear the OCxREF signal on an external event.</li> <li>• OC5CE and OC6CE are not available for all F3 devices</li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 OC1CE LL_TIM_OC_EnableClear</li> <li>• CCMR1 OC2CE LL_TIM_OC_EnableClear</li> <li>• CCMR2 OC3CE LL_TIM_OC_EnableClear</li> </ul>                                                                                                                                                                                                                                                                      |

- CCMR2 OC4CE LL\_TIM\_OC\_EnableClear
- CCMR3 OC5CE LL\_TIM\_OC\_EnableClear
- CCMR3 OC6CE LL\_TIM\_OC\_EnableClear

### LL\_TIM\_OC\_DisableClear

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_OC_DisableClear(<br/>    TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                 |
| Function description                              | Disable clearing the output channel on an external event.                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_OCXREF_CLEAR_INSTANCE(TIMx) can be used to check whether or not a timer instance can clear the OCxREF signal on an external event.</li> <li>• OC5CE and OC6CE are not available for all F3 devices</li> <li>• CH5 and CH6 channels are not available for all F3 devices</li> </ul>                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 OC1CE LL_TIM_OC_DisableClear</li> <li>• CCMR1 OC2CE LL_TIM_OC_DisableClear</li> <li>• CCMR2 OC3CE LL_TIM_OC_DisableClear</li> <li>• CCMR2 OC4CE LL_TIM_OC_DisableClear</li> <li>• CCMR3 OC5CE LL_TIM_OC_DisableClear</li> <li>• CCMR3 OC6CE LL_TIM_OC_DisableClear</li> </ul>                                                                            |

### LL\_TIM\_OC\_IsEnabledClear

|                      |                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_TIM_OC_IsEnabledClear(<br/>    TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                           |
| Function description | Indicates clearing the output channel on an external event is enabled for the output channel.                                                                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• This function enables clearing the output channel on an external event.</li> <li>• This function can only be used in Output compare and PWM</li> </ul>                                                                                                                                                                                                         |

Reference Manual to  
LL API cross  
reference:

- modes. It does not work in Forced mode.
- Macro IS\_TIM\_OCXREF\_CLEAR\_INSTANCE(TIMx) can be used to check whether or not a timer instance can clear the OCxREF signal on an external event.
- OC5CE and OC6CE are not available for all F3 devices
- CH5 and CH6 channels are not available for all F3 devices
- CCMR1 OC1CE LL\_TIM\_OC\_IsEnabledClear
- CCMR1 OC2CE LL\_TIM\_OC\_IsEnabledClear
- CCMR2 OC3CE LL\_TIM\_OC\_IsEnabledClear
- CCMR2 OC4CE LL\_TIM\_OC\_IsEnabledClear
- CCMR3 OC5CE LL\_TIM\_OC\_IsEnabledClear
- CCMR3 OC6CE LL\_TIM\_OC\_IsEnabledClear

### **LL\_TIM\_OC\_SetDeadTime**

|                                                   |                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_OC_SetDeadTime(<br/>TIM_TypeDef * TIMx, uint32_t DeadTime)</code></b>                                                                                                                                                                              |
| Function description                              | Set the dead-time delay (delay inserted between the rising edge of the OCxREF signal and the rising edge if the Ocx and OCxN signals).                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>DeadTime:</b> between Min_Data=0 and Max_Data=255</li> </ul>                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not dead-time insertion feature is supported by a timer instance.</li> <li>• Helper macro __LL_TIM_CALC_DEADTIME can be used to calculate the DeadTime parameter</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDTR DTG LL_TIM_OC_SetDeadTime</li> </ul>                                                                                                                                                                                                     |

### **LL\_TIM\_OC\_SetCompareCH1**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_TIM_OC_SetCompareCH1(<br/>TIM_TypeDef * TIMx, uint32_t CompareValue)</code></b>                                                                                                                                                                                                                                                                                         |
| Function description | Set compare value for output channel 1 (TIMx_CCR1).                                                                                                                                                                                                                                                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>CompareValue:</b> between Min_Data=0 and Max_Data=65535</li> </ul>                                                                                                                                                                                                                                                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                         |
| Notes                | <ul style="list-style-type: none"> <li>• In 32-bit timer implementations compare value can be between 0x00000000 and 0xFFFFFFFF.</li> <li>• Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> <li>• Macro IS_TIM_CC1_INSTANCE(TIMx) can be used to check whether or not output channel 1 is supported by a timer</li> </ul> |

instance.

Reference Manual to  
LL API cross  
reference:

- CCR1 CCR1 LL\_TIM\_OC\_SetCompareCH1

### **LL\_TIM\_OC\_SetCompareCH2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_OC_SetCompareCH2(<br/>    TIM_TypeDef * TIMx, uint32_t CompareValue)</code>                                                                                                                                                                                                                                                                                                     |
| Function description                              | Set compare value for output channel 2 (TIMx_CCR2).                                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>CompareValue:</b> between Min_Data=0 and Max_Data=65535</li> </ul>                                                                                                                                                                                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>• In 32-bit timer implementations compare value can be between 0x00000000 and 0xFFFFFFFF.</li> <li>• Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> <li>• Macro IS_TIM_CC2_INSTANCE(TIMx) can be used to check whether or not output channel 2 is supported by a timer instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR2 CCR2 LL_TIM_OC_SetCompareCH2</li> </ul>                                                                                                                                                                                                                                                                                                                             |

### **LL\_TIM\_OC\_SetCompareCH3**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_OC_SetCompareCH3(<br/>    TIM_TypeDef * TIMx, uint32_t CompareValue)</code>                                                                                                                                                                                                                                                                                                   |
| Function description                              | Set compare value for output channel 3 (TIMx_CCR3).                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>CompareValue:</b> between Min_Data=0 and Max_Data=65535</li> </ul>                                                                                                                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>• In 32-bit timer implementations compare value can be between 0x00000000 and 0xFFFFFFFF.</li> <li>• Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> <li>• Macro IS_TIM_CC3_INSTANCE(TIMx) can be used to check whether or not output channel is supported by a timer instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR3 CCR3 LL_TIM_OC_SetCompareCH3</li> </ul>                                                                                                                                                                                                                                                                                                                           |

**LL\_TIM\_OC\_SetCompareCH4**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_TIM_OC_SetCompareCH4<br/>(TIM_TypeDef * TIMx, uint32_t CompareValue)</code>                                                                                                                                                                                                                                                                                                          |
| Function description                              | Set compare value for output channel 4 (TIMx_CCR4).                                                                                                                                                                                                                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>CompareValue:</b> between Min_Data=0 and Max_Data=65535</li> </ul>                                                                                                                                                                                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>• In 32-bit timer implementations compare value can be between 0x00000000 and 0xFFFFFFFF.</li> <li>• Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> <li>• Macro IS_TIM_CC4_INSTANCE(TIMx) can be used to check whether or not output channel 4 is supported by a timer instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR4 CCR4 LL_TIM_OC_SetCompareCH4</li> </ul>                                                                                                                                                                                                                                                                                                                             |

**LL\_TIM\_OC\_SetCompareCH5**

|                                                   |                                                                                                                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_TIM_OC_SetCompareCH5<br/>(TIM_TypeDef * TIMx, uint32_t CompareValue)</code>                                                                                                                              |
| Function description                              | Set compare value for output channel 5 (TIMx_CCR5).                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>CompareValue:</b> between Min_Data=0 and Max_Data=65535</li> </ul>                                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_CC5_INSTANCE(TIMx) can be used to check whether or not output channel 5 is supported by a timer instance.</li> <li>• CH5 channel is not available for all F3 devices</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR5 CCR5 LL_TIM_OC_SetCompareCH5</li> </ul>                                                                                                                                                 |

**LL\_TIM\_OC\_SetCompareCH6**

|                      |                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_TIM_OC_SetCompareCH6<br/>(TIM_TypeDef * TIMx, uint32_t CompareValue)</code>                                              |
| Function description | Set compare value for output channel 6 (TIMx_CCR6).                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>CompareValue:</b> between Min_Data=0 and Max_Data=65535</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                       |

- Notes
- Macro IS\_TIM\_CC6\_INSTANCE(TIMx) can be used to check whether or not output channel 6 is supported by a timer instance. CCR6 CCR6 LL\_TIM\_OC\_SetCompareCH6
  - CH6 channel is not available for all F3 devices

### **LL\_TIM\_OC\_GetCompareCH1**

- Function name **STATIC\_INLINE uint32\_t LL\_TIM\_OC\_GetCompareCH1 (TIM\_TypeDef \* TIMx)**
- Function description Get compare value (TIMx\_CCR1) set for output channel 1.
- Parameters
- TIMx:** Timer instance
- Return values
- CompareValue:** (between Min\_Data=0 and Max\_Data=65535)
- Notes
- In 32-bit timer implementations returned compare value can be between 0x00000000 and 0xFFFFFFFF.
  - Macro IS\_TIM\_32B\_COUNTER\_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.
  - Macro IS\_TIM\_CC1\_INSTANCE(TIMx) can be used to check whether or not output channel 1 is supported by a timer instance.
- Reference Manual to LL API cross reference:
- CCR1 CCR1 LL\_TIM\_OC\_GetCompareCH1

### **LL\_TIM\_OC\_GetCompareCH2**

- Function name **STATIC\_INLINE uint32\_t LL\_TIM\_OC\_GetCompareCH2 (TIM\_TypeDef \* TIMx)**
- Function description Get compare value (TIMx\_CCR2) set for output channel 2.
- Parameters
- TIMx:** Timer instance
- Return values
- CompareValue:** (between Min\_Data=0 and Max\_Data=65535)
- Notes
- In 32-bit timer implementations returned compare value can be between 0x00000000 and 0xFFFFFFFF.
  - Macro IS\_TIM\_32B\_COUNTER\_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.
  - Macro IS\_TIM\_CC2\_INSTANCE(TIMx) can be used to check whether or not output channel 2 is supported by a timer instance.
- Reference Manual to LL API cross reference:
- CCR2 CCR2 LL\_TIM\_OC\_GetCompareCH2

### **LL\_TIM\_OC\_GetCompareCH3**

- Function name **STATIC\_INLINE uint32\_t LL\_TIM\_OC\_GetCompareCH3 (TIM\_TypeDef \* TIMx)**



|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function description                              | Get compare value (TIMx_CCR3) set for output channel 3.                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>CompareValue:</b> (between Min_Data=0 and Max_Data=65535)</li> </ul>                                                                                                                                                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• In 32-bit timer implementations returned compare value can be between 0x00000000 and 0xFFFFFFFF.</li> <li>• Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> <li>• Macro IS_TIM_CC3_INSTANCE(TIMx) can be used to check whether or not output channel 3 is supported by a timer instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR3 CCR3 LL_TIM_OC_GetCompareCH3</li> </ul>                                                                                                                                                                                                                                                                                                                                      |

### LL\_TIM\_OC\_GetCompareCH4

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_OC_GetCompareCH4(<br/>(TIM_TypeDef * TIMx)</code>                                                                                                                                                                                                                                                                                                                                    |
| Function description                              | Get compare value (TIMx_CCR4) set for output channel 4.                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>CompareValue:</b> (between Min_Data=0 and Max_Data=65535)</li> </ul>                                                                                                                                                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>• In 32-bit timer implementations returned compare value can be between 0x00000000 and 0xFFFFFFFF.</li> <li>• Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> <li>• Macro IS_TIM_CC4_INSTANCE(TIMx) can be used to check whether or not output channel 4 is supported by a timer instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR4 CCR4 LL_TIM_OC_GetCompareCH4</li> </ul>                                                                                                                                                                                                                                                                                                                                      |

### LL\_TIM\_OC\_GetCompareCH5

|                      |                                                                                                                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_TIM_OC_GetCompareCH5(<br/>(TIM_TypeDef * TIMx)</code>                                                                                                                      |
| Function description | Get compare value (TIMx_CCR5) set for output channel 5.                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>CompareValue:</b> (between Min_Data=0 and Max_Data=65535)</li> </ul>                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_TIM_CC5_INSTANCE(TIMx) can be used to check whether or not output channel 5 is supported by a timer instance. CCR5 CCR5 LL_TIM_OC_GetCompareCH5</li> </ul> |

- CH5 channel is not available for all F3 devices

### **LL\_TIM\_OC\_GetCompareCH6**

|                      |                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_TIM_OC_GetCompareCH6(<br/>    TIM_TypeDef * TIMx)</code>                                                                                                                                                                              |
| Function description | Get compare value (TIMx_CCR6) set for output channel 6.                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                         |
| Return values        | <ul style="list-style-type: none"> <li>• <b>CompareValue:</b> (between Min_Data=0 and Max_Data=65535)</li> </ul>                                                                                                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_TIM_CC6_INSTANCE(TIMx) can be used to check whether or not output channel 6 is supported by a timer instance. CCR6 CCR6 LL_TIM_OC_GetCompareCH6</li> <li>• CH6 channel is not available for all F3 devices</li> </ul> |

### **LL\_TIM\_SetCH5CombinedChannels**

|                      |                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_SetCH5CombinedChannels(<br/>    TIM_TypeDef * TIMx, uint32_t GroupCH5)</code>                                                                                                                                                                                                                                                                      |
| Function description | Select on which reference signal the OC5REF is combined to.                                                                                                                                                                                                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>GroupCH5:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_TIM_GROUPCH5_NONE</li> <li>- LL_TIM_GROUPCH5_OC1REFC</li> <li>- LL_TIM_GROUPCH5_OC2REFC</li> <li>- LL_TIM_GROUPCH5_OC3REFC</li> </ul> </li> </ul>                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_TIM_COMBINED3PHASEPWM_INSTANCE(TIMx) can be used to check whether or not a timer instance supports the combined 3-phase PWM mode. CCR5 GC5C3 LL_TIM_SetCH5CombinedChannels CCR5 GC5C2 LL_TIM_SetCH5CombinedChannels CCR5 GC5C1 LL_TIM_SetCH5CombinedChannels</li> <li>• CH5 channel is not available for all F3 devices</li> </ul> |

### **LL\_TIM\_IC\_Config**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_IC_Config (TIM_TypeDef *<br/>    TIMx, uint32_t Channel, uint32_t Configuration)</code>                                                                                                                                                                                                                                                                                                |
| Function description | Configure input channel.                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH4</li> </ul> </li> <li>• <b>Configuration:</b> This parameter must be a combination of all</li> </ul> |

the following values:

- LL\_TIM\_ACTIVEINPUT\_DIRECTTI or  
LL\_TIM\_ACTIVEINPUT\_INDIRECTTI or  
LL\_TIM\_ACTIVEINPUT\_TRC
- LL\_TIM\_ICPSC\_DIV1 or ... or LL\_TIM\_ICPSC\_DIV8
- LL\_TIM\_IC\_FILTER\_FDIV1 or ... or  
LL\_TIM\_IC\_FILTER\_FDIV32\_N8
- LL\_TIM\_IC\_POLARITY\_RISING or  
LL\_TIM\_IC\_POLARITY\_FALLING or  
LL\_TIM\_IC\_POLARITY\_BOTHEDGE

#### Return values

Reference Manual to  
LL API cross  
reference:

- **None**
- CCMR1 CC1S LL\_TIM\_IC\_Config
- CCMR1 IC1PSC LL\_TIM\_IC\_Config
- CCMR1 IC1F LL\_TIM\_IC\_Config
- CCMR1 CC2S LL\_TIM\_IC\_Config
- CCMR1 IC2PSC LL\_TIM\_IC\_Config
- CCMR1 IC2F LL\_TIM\_IC\_Config
- CCMR2 CC3S LL\_TIM\_IC\_Config
- CCMR2 IC3PSC LL\_TIM\_IC\_Config
- CCMR2 IC3F LL\_TIM\_IC\_Config
- CCMR2 CC4S LL\_TIM\_IC\_Config
- CCMR2 IC4PSC LL\_TIM\_IC\_Config
- CCMR2 IC4F LL\_TIM\_IC\_Config
- CCER CC1P LL\_TIM\_IC\_Config
- CCER CC1NP LL\_TIM\_IC\_Config
- CCER CC2P LL\_TIM\_IC\_Config
- CCER CC2NP LL\_TIM\_IC\_Config
- CCER CC3P LL\_TIM\_IC\_Config
- CCER CC3NP LL\_TIM\_IC\_Config
- CCER CC4P LL\_TIM\_IC\_Config
- CCER CC4NP LL\_TIM\_IC\_Config

## LL\_TIM\_IC\_SetActiveInput

#### Function name

```
STATIC_INLINE void LL_TIM_IC_SetActiveInput
(TIM_TypeDef * TIMx, uint32_t Channel, uint32_t
ICActiveInput)
```

#### Function description

Set the active input.

#### Parameters

- **TIMx:** Timer instance
- **Channel:** This parameter can be one of the following values:
  - LL\_TIM\_CHANNEL\_CH1
  - LL\_TIM\_CHANNEL\_CH2
  - LL\_TIM\_CHANNEL\_CH3
  - LL\_TIM\_CHANNEL\_CH4
- **ICActiveInput:** This parameter can be one of the following values:
  - LL\_TIM\_ACTIVEINPUT\_DIRECTTI
  - LL\_TIM\_ACTIVEINPUT\_INDIRECTTI
  - LL\_TIM\_ACTIVEINPUT\_TRC

#### Return values

- **None**

- Reference Manual to  
LL API cross  
reference:
- CCMR1 CC1S LL\_TIM\_IC\_SetActiveInput
  - CCMR1 CC2S LL\_TIM\_IC\_SetActiveInput
  - CCMR2 CC3S LL\_TIM\_IC\_SetActiveInput
  - CCMR2 CC4S LL\_TIM\_IC\_SetActiveInput

### **LL\_TIM\_IC\_GetActiveInput**

|                                                   |                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_TIM_IC_GetActiveInput(<br/>(TIM_TypeDef * TIMx, uint32_t Channel)</code></b>                                                                                                                                                                                                                |
| Function description                              | Get the current active input.                                                                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH4</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_ACTIVEINPUT_DIRECTTI</li> <li>- LL_TIM_ACTIVEINPUT_INDIRECTTI</li> <li>- LL_TIM_ACTIVEINPUT_TRC</li> </ul> </li> </ul>                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 CC1S LL_TIM_IC_GetActiveInput</li> <li>• CCMR1 CC2S LL_TIM_IC_GetActiveInput</li> <li>• CCMR2 CC3S LL_TIM_IC_GetActiveInput</li> <li>• CCMR2 CC4S LL_TIM_IC_GetActiveInput</li> </ul>                                                                                            |

### **LL\_TIM\_IC\_SetPrescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_IC_SetPrescaler(<br/>(TIM_TypeDef * TIMx, uint32_t Channel, uint32_t ICPrescaler)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Set the prescaler of input channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH4</li> </ul> </li> <li>• <b>ICPrescaler:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_ICPSC_DIV1</li> <li>- LL_TIM_ICPSC_DIV2</li> <li>- LL_TIM_ICPSC_DIV4</li> <li>- LL_TIM_ICPSC_DIV8</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 IC1PSC LL_TIM_IC_SetPrescaler</li> <li>• CCMR1 IC2PSC LL_TIM_IC_SetPrescaler</li> <li>• CCMR2 IC3PSC LL_TIM_IC_SetPrescaler</li> <li>• CCMR2 IC4PSC LL_TIM_IC_SetPrescaler</li> </ul>                                                                                                                                                                                                                                                                                                                                            |

**LL\_TIM\_IC\_GetPrescaler**

|                                                   |                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IC_GetPrescaler(<br/>    TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                                                                                                 |
| Function description                              | Get the current prescaler value acting on an input channel.                                                                                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH4</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_TIM_ICPSC_DIV1</li> <li>- LL_TIM_ICPSC_DIV2</li> <li>- LL_TIM_ICPSC_DIV4</li> <li>- LL_TIM_ICPSC_DIV8</li> </ul> </li> </ul>                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 IC1PSC LL_TIM_IC_GetPrescaler</li> <li>• CCMR1 IC2PSC LL_TIM_IC_GetPrescaler</li> <li>• CCMR2 IC3PSC LL_TIM_IC_GetPrescaler</li> <li>• CCMR2 IC4PSC LL_TIM_IC_GetPrescaler</li> </ul>                                                                                                        |

**LL\_TIM\_IC\_SetFilter**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_IC_SetFilter (TIM_TypeDef *<br/>    TIMx, uint32_t Channel, uint32_t ICFilter)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description | Set the input filter duration.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH4</li> </ul> </li> <li>• <b>ICFilter:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_TIM_IC_FILTER_FDIV1</li> <li>- LL_TIM_IC_FILTER_FDIV1_N2</li> <li>- LL_TIM_IC_FILTER_FDIV1_N4</li> <li>- LL_TIM_IC_FILTER_FDIV1_N8</li> <li>- LL_TIM_IC_FILTER_FDIV2_N6</li> <li>- LL_TIM_IC_FILTER_FDIV2_N8</li> <li>- LL_TIM_IC_FILTER_FDIV4_N6</li> <li>- LL_TIM_IC_FILTER_FDIV4_N8</li> <li>- LL_TIM_IC_FILTER_FDIV8_N6</li> <li>- LL_TIM_IC_FILTER_FDIV8_N8</li> <li>- LL_TIM_IC_FILTER_FDIV16_N5</li> <li>- LL_TIM_IC_FILTER_FDIV16_N6</li> <li>- LL_TIM_IC_FILTER_FDIV16_N8</li> <li>- LL_TIM_IC_FILTER_FDIV32_N5</li> <li>- LL_TIM_IC_FILTER_FDIV32_N6</li> <li>- LL_TIM_IC_FILTER_FDIV32_N8</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                  |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 IC1F LL_TIM_IC_SetFilter</li> <li>• CCMR1 IC2F LL_TIM_IC_SetFilter</li> <li>• CCMR2 IC3F LL_TIM_IC_SetFilter</li> <li>• CCMR2 IC4F LL_TIM_IC_SetFilter</li> </ul> |

### LL\_TIM\_IC\_GetFilter

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IC_GetFilter (TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Function description                              | Get the input filter duration.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH4</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_IC_FILTER_FDIV1</li> <li>- LL_TIM_IC_FILTER_FDIV1_N2</li> <li>- LL_TIM_IC_FILTER_FDIV1_N4</li> <li>- LL_TIM_IC_FILTER_FDIV1_N8</li> <li>- LL_TIM_IC_FILTER_FDIV2_N6</li> <li>- LL_TIM_IC_FILTER_FDIV2_N8</li> <li>- LL_TIM_IC_FILTER_FDIV4_N6</li> <li>- LL_TIM_IC_FILTER_FDIV4_N8</li> <li>- LL_TIM_IC_FILTER_FDIV8_N6</li> <li>- LL_TIM_IC_FILTER_FDIV8_N8</li> <li>- LL_TIM_IC_FILTER_FDIV16_N5</li> <li>- LL_TIM_IC_FILTER_FDIV16_N6</li> <li>- LL_TIM_IC_FILTER_FDIV16_N8</li> <li>- LL_TIM_IC_FILTER_FDIV32_N5</li> <li>- LL_TIM_IC_FILTER_FDIV32_N6</li> <li>- LL_TIM_IC_FILTER_FDIV32_N8</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCMR1 IC1F LL_TIM_IC_SetFilter</li> <li>• CCMR1 IC2F LL_TIM_IC_SetFilter</li> <li>• CCMR2 IC3F LL_TIM_IC_SetFilter</li> <li>• CCMR2 IC4F LL_TIM_IC_SetFilter</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

### LL\_TIM\_IC\_SetPolarity

|                      |                                                                                                                                                                                                                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_IC_SetPolarity (TIM_TypeDef * TIMx, uint32_t Channel, uint32_t IC_Polarity)</code>                                                                                                                                                                              |
| Function description | Set the input channel polarity.                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> </ul> </li> </ul> |

- LL\_TIM\_CHANNEL\_CH4
- **ICPolarity:** This parameter can be one of the following values:
  - LL\_TIM\_IC\_POLARITY\_RISING
  - LL\_TIM\_IC\_POLARITY\_FALLING
  - LL\_TIM\_IC\_POLARITY\_BOTHEDGE
- **Return values**
- Reference Manual to LL API cross reference:
  - CCER CC1P LL\_TIM\_IC\_SetPolarity
  - CCER CC1NP LL\_TIM\_IC\_SetPolarity
  - CCER CC2P LL\_TIM\_IC\_SetPolarity
  - CCER CC2NP LL\_TIM\_IC\_SetPolarity
  - CCER CC3P LL\_TIM\_IC\_SetPolarity
  - CCER CC3NP LL\_TIM\_IC\_SetPolarity
  - CCER CC4P LL\_TIM\_IC\_SetPolarity
  - CCER CC4NP LL\_TIM\_IC\_SetPolarity

### **LL\_TIM\_IC\_GetPolarity**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_TIM_IC_GetPolarity(<br/>  TIM_TypeDef * TIMx, uint32_t Channel)</code>                                                                                                                                                                                                                                                                                             |
| Function description                        | Get the current input channel polarity.                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_TIM_CHANNEL_CH1</li> <li>- LL_TIM_CHANNEL_CH2</li> <li>- LL_TIM_CHANNEL_CH3</li> <li>- LL_TIM_CHANNEL_CH4</li> </ul> </li> </ul>                                                            |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_TIM_IC_POLARITY_RISING</li> <li>- LL_TIM_IC_POLARITY_FALLING</li> <li>- LL_TIM_IC_POLARITY_BOTHEDGE</li> </ul> </li> </ul>                                                                                                                 |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CCER CC1P LL_TIM_IC_GetPolarity</li> <li>• CCER CC1NP LL_TIM_IC_GetPolarity</li> <li>• CCER CC2P LL_TIM_IC_GetPolarity</li> <li>• CCER CC2NP LL_TIM_IC_GetPolarity</li> <li>• CCER CC3P LL_TIM_IC_GetPolarity</li> <li>• CCER CC3NP LL_TIM_IC_GetPolarity</li> <li>• CCER CC4P LL_TIM_IC_GetPolarity</li> <li>• CCER CC4NP LL_TIM_IC_GetPolarity</li> </ul> |

### **LL\_TIM\_IC\_EnableXORCombination**

|                      |                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_IC_EnableXORCombination(<br/>  TIM_TypeDef * TIMx)</code> |
| Function description | Connect the TIMx_CH1, CH2 and CH3 pins to the TI1 input (XOR combination).                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                             |

---

|                                             |                                                                                                                                                               |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_TIM_XOR_INSTANCE(TIMx) can be used to check whether or not a timer instance provides an XOR input.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR2 TI1S LL_TIM_IC_EnableXORCombination</li> </ul>                                                                     |

### LL\_TIM\_IC\_DisableXORCombination

|                                             |                                                                                                                                                               |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE void LL_TIM_IC_DisableXORCombination (TIM_TypeDef * TIMx)</code></b>                                                                 |
| Function description                        | Disconnect the TIMx_CH1, CH2 and CH3 pins from the TI1 input.                                                                                                 |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer instance</li> </ul>                                                                                 |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                 |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_TIM_XOR_INSTANCE(TIMx) can be used to check whether or not a timer instance provides an XOR input.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR2 TI1S LL_TIM_IC_DisableXORCombination</li> </ul>                                                                    |

### LL\_TIM\_IC\_IsEnabledXORCombination

|                                             |                                                                                                                                                               |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>__STATIC_INLINE uint32_t LL_TIM_IC_IsEnabledXORCombination (TIM_TypeDef * TIMx)</code></b>                                                           |
| Function description                        | Indicates whether the TIMx_CH1, CH2 and CH3 pins are connected to the TI1 input.                                                                              |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer instance</li> </ul>                                                                                 |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                              |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_TIM_XOR_INSTANCE(TIMx) can be used to check whether or not a timer instance provides an XOR input.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR2 TI1S LL_TIM_IC_IsEnabledXORCombination</li> </ul>                                                                  |

### LL\_TIM\_IC\_GetCaptureCH1

|                      |                                                                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_TIM_IC_GetCaptureCH1 (TIM_TypeDef * TIMx)</code></b>                                                                                                                                                                                             |
| Function description | Get captured value for input channel 1.                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                                         |
| Return values        | <ul style="list-style-type: none"> <li><b>CapturedValue:</b> (between Min_Data=0 and Max_Data=65535)</li> </ul>                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>In 32-bit timer implementations returned captured value can be between 0x00000000 and 0xFFFFFFFF.</li> <li>Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> </ul> |

- Macro IS\_TIM\_CC1\_INSTANCE(TIMx) can be used to check whether or not input channel 1 is supported by a timer instance.

Reference Manual to  
LL API cross  
reference:

- CCR1 CCR1 LL\_TIM\_IC\_GetCaptureCH1

### **LL\_TIM\_IC\_GetCaptureCH2**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IC_GetCaptureCH2(<br/>                  TIM_TypeDef * TIMx)</code>                                                                                                                                                                                                                                                                                                                   |
| Function description                              | Get captured value for input channel 2.                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>CapturedValue:</b> (between Min_Data=0 and Max_Data=65535)</li> </ul>                                                                                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• In 32-bit timer implementations returned captured value can be between 0x00000000 and 0xFFFFFFFF.</li> <li>• Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> <li>• Macro IS_TIM_CC2_INSTANCE(TIMx) can be used to check whether or not input channel 2 is supported by a timer instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR2 CCR2 LL_TIM_IC_GetCaptureCH2</li> </ul>                                                                                                                                                                                                                                                                                                                                      |

### **LL\_TIM\_IC\_GetCaptureCH3**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IC_GetCaptureCH3(<br/>                  TIM_TypeDef * TIMx)</code>                                                                                                                                                                                                                                                                                                                   |
| Function description                              | Get captured value for input channel 3.                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>CapturedValue:</b> (between Min_Data=0 and Max_Data=65535)</li> </ul>                                                                                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• In 32-bit timer implementations returned captured value can be between 0x00000000 and 0xFFFFFFFF.</li> <li>• Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> <li>• Macro IS_TIM_CC3_INSTANCE(TIMx) can be used to check whether or not input channel 3 is supported by a timer instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR3 CCR3 LL_TIM_IC_GetCaptureCH3</li> </ul>                                                                                                                                                                                                                                                                                                                                      |

**LL\_TIM\_IC\_GetCaptureCH4**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IC_GetCaptureCH4(<br/>    TIM_TypeDef * TIMx)</code>                                                                                                                                                                                                                                                                                                                                 |
| Function description                              | Get captured value for input channel 4.                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                                                                                                                                                                            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>CapturedValue:</b> (between Min_Data=0 and Max_Data=65535)</li> </ul>                                                                                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• In 32-bit timer implementations returned captured value can be between 0x00000000 and 0xFFFFFFFF.</li> <li>• Macro IS_TIM_32B_COUNTER_INSTANCE(TIMx) can be used to check whether or not a timer instance supports a 32 bits counter.</li> <li>• Macro IS_TIM_CC4_INSTANCE(TIMx) can be used to check whether or not input channel 4 is supported by a timer instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CCR4 CCR4 LL_TIM_IC_GetCaptureCH4</li> </ul>                                                                                                                                                                                                                                                                                                                                      |

**LL\_TIM\_EnableExternalClock**

|                                                   |                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableExternalClock(<br/>    TIM_TypeDef * TIMx)</code>                                                                                                                                                                                                               |
| Function description                              | Enable external clock mode 2.                                                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• When external clock mode 2 is enabled the counter is clocked by any active edge on the ETRF signal.</li> <li>• Macro IS_TIM_CLOCKSOURCE_ETRMODE2_INSTANCE(TIMx) can be used to check whether or not a timer instance supports external clock mode2.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SMCR ECE LL_TIM_EnableExternalClock</li> </ul>                                                                                                                                                                                                                 |

**LL\_TIM\_DisableExternalClock**

|                      |                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_DisableExternalClock(<br/>    TIM_TypeDef * TIMx)</code>                                                                |
| Function description | Disable external clock mode 2.                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_TIM_CLOCKSOURCE_ETRMODE2_INSTANCE(TIMx) can be used to check whether or not a timer instance</li> </ul> |

supports external clock mode2.

Reference Manual to  
LL API cross  
reference:

- SMCR ECE LL\_TIM\_DisableExternalClock

### **LL\_TIM\_IsEnabledExternalClock**

Function name **\_STATIC\_INLINE uint32\_t LL\_TIM\_IsEnabledExternalClock  
(TIM\_TypeDef \* TIMx)**

Function description Indicate whether external clock mode 2 is enabled.

Parameters

- **TIMx:** Timer instance

Return values

- **State:** of bit (1 or 0).

Notes

- Macro  
`IS_TIM_CLOCKSOURCE_ETRMODE2_INSTANCE(TIMx)`  
can be used to check whether or not a timer instance  
supports external clock mode2.

Reference Manual to  
LL API cross  
reference:

- SMCR ECE LL\_TIM\_IsEnabledExternalClock

### **LL\_TIM\_SetClockSource**

Function name **\_STATIC\_INLINE void LL\_TIM\_SetClockSource  
(TIM\_TypeDef \* TIMx, uint32\_t ClockSource)**

Function description Set the clock source of the counter clock.

Parameters

- **TIMx:** Timer instance
- **ClockSource:** This parameter can be one of the following values:
  - `LL_TIM_CLOCKSOURCE_INTERNAL`
  - `LL_TIM_CLOCKSOURCE_EXT_MODE1`
  - `LL_TIM_CLOCKSOURCE_EXT_MODE2`

Return values

- **None**

Notes

- when selected clock source is external clock mode 1, the timer input the external clock is applied is selected by calling the `LL_TIM_SetTriggerInput()` function. This timer input must be configured by calling the `LL_TIM_IC_Config()` function.
- Macro  
`IS_TIM_CLOCKSOURCE_ETRMODE1_INSTANCE(TIMx)`  
can be used to check whether or not a timer instance  
supports external clock mode1.
- Macro  
`IS_TIM_CLOCKSOURCE_ETRMODE2_INSTANCE(TIMx)`  
can be used to check whether or not a timer instance  
supports external clock mode2.

Reference Manual to  
LL API cross  
reference:

- SMCR SMS LL\_TIM\_SetClockSource
- SMCR ECE LL\_TIM\_SetClockSource

**LL\_TIM\_SetEncoderMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_SetEncoderMode<br/>(TIM_TypeDef * TIMx, uint32_t EncoderMode)</code>                                                                                                                                                                                                                                                             |
| Function description                              | Set the encoder interface mode.                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>EncoderMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_TIM_ENCODERMODE_X2_TI1</code></li> <li>- <code>LL_TIM_ENCODERMODE_X2_TI2</code></li> <li>- <code>LL_TIM_ENCODERMODE_X4_TI12</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro <code>IS_TIM_ENCODER_INTERFACE_INSTANCE(TIMx)</code> can be used to check whether or not a timer instance supports the encoder mode.</li> </ul>                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>SMCR SMS LL_TIM_SetEncoderMode</code></li> </ul>                                                                                                                                                                                                                                                                    |

**LL\_TIM\_SetTriggerOutput**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_SetTriggerOutput<br/>(TIM_TypeDef * TIMx, uint32_t TimerSynchronization)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Set the trigger output (TRGO) used for timer synchronization .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>TimerSynchronization:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- <code>LL_TIM_TRGO_RESET</code></li> <li>- <code>LL_TIM_TRGO_ENABLE</code></li> <li>- <code>LL_TIM_TRGO_UPDATE</code></li> <li>- <code>LL_TIM_TRGO_CC1IF</code></li> <li>- <code>LL_TIM_TRGO_OC1REF</code></li> <li>- <code>LL_TIM_TRGO_OC2REF</code></li> <li>- <code>LL_TIM_TRGO_OC3REF</code></li> <li>- <code>LL_TIM_TRGO_OC4REF</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro <code>IS_TIM_MASTER_INSTANCE(TIMx)</code> can be used to check whether or not a timer instance can operate as a master timer.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• <code>CR2 MMS LL_TIM_SetTriggerOutput</code></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

**LL\_TIM\_SetTriggerOutput2**

|                      |                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_SetTriggerOutput2<br/>(TIM_TypeDef * TIMx, uint32_t ADCSynchronization)</code> |
| Function description | Set the trigger output 2 (TRGO2) used for ADC synchronization .                                                  |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer Instance</li> <li>• <b>ADC Synchronization:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_TIM_TRGO2_RESET</li> <li>– LL_TIM_TRGO2_ENABLE</li> <li>– LL_TIM_TRGO2_UPDATE</li> <li>– LL_TIM_TRGO2_CC1F</li> <li>– LL_TIM_TRGO2_OC1</li> <li>– LL_TIM_TRGO2_OC2</li> <li>– LL_TIM_TRGO2_OC3</li> <li>– LL_TIM_TRGO2_OC4</li> <li>– LL_TIM_TRGO2_OC5</li> <li>– LL_TIM_TRGO2_OC6</li> <li>– LL_TIM_TRGO2_OC4_RISINGFALLING</li> <li>– LL_TIM_TRGO2_OC6_RISINGFALLING</li> <li>– LL_TIM_TRGO2_OC4_RISING_OC6_RISING</li> <li>– LL_TIM_TRGO2_OC4_RISING_OC6_FALLING</li> <li>– LL_TIM_TRGO2_OC5_RISING_OC6_RISING</li> <li>– LL_TIM_TRGO2_OC5_RISING_OC6_FALLING</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_TRGO2_INSTANCE(TIMx) can be used to check whether or not a timer instance can be used for ADC synchronization.</li> <li>• OC5 and OC6 are not available for all F3 devices</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 MMS2 LL_TIM_SetTriggerOutput2</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

## LL\_TIM\_SetSlaveMode

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_SetSlaveMode (TIM_TypeDef *<br/>TIMx, uint32_t SlaveMode)</code>                                                                                                                                                                                                                                                                                                                 |
| Function description                              | Set the synchronization mode of a slave timer.                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>SlaveMode:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_TIM_SLAVEMODE_DISABLED</li> <li>– LL_TIM_SLAVEMODE_RESET</li> <li>– LL_TIM_SLAVEMODE_GATED</li> <li>– LL_TIM_SLAVEMODE_TRIGGER</li> <li>– LL_TIM_SLAVEMODE_COMBINED_RESETTRIGGER</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_SLAVE_INSTANCE(TIMx) can be used to check whether or not a timer instance can operate as a slave timer.</li> </ul>                                                                                                                                                                                                                                           |
| Reference Manual<br>to LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SMCR SMS LL_TIM_SetSlaveMode</li> </ul>                                                                                                                                                                                                                                                                                                                                   |

**LL\_TIM\_SetTriggerInput**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_SetTriggerInput (TIM_TypeDef * TIMx, uint32_t TriggerInput)</code>                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Set the selects the trigger input to be used to synchronize the counter.                                                                                                                                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>TriggerInput:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_TS_ITR0</li> <li>– LL_TIM_TS_ITR1</li> <li>– LL_TIM_TS_ITR2</li> <li>– LL_TIM_TS_ITR3</li> <li>– LL_TIM_TS_TI1F_ED</li> <li>– LL_TIM_TS_TI1FP1</li> <li>– LL_TIM_TS_TI2FP2</li> <li>– LL_TIM_TS_ETRF</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_SLAVE_INSTANCE(TIMx) can be used to check whether or not a timer instance can operate as a slave timer.</li> </ul>                                                                                                                                                                                                                                                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SMCR TS LL_TIM_SetTriggerInput</li> </ul>                                                                                                                                                                                                                                                                                                                                                  |

**LL\_TIM\_EnableMasterSlaveMode**

|                                                   |                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableMasterSlaveMode (TIM_TypeDef * TIMx)</code>                                                                                      |
| Function description                              | Enable the Master/Slave mode.                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_SLAVE_INSTANCE(TIMx) can be used to check whether or not a timer instance can operate as a slave timer.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SMCR MSM LL_TIM_EnableMasterSlaveMode</li> </ul>                                                                                |

**LL\_TIM\_DisableMasterSlaveMode**

|                      |                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_DisableMasterSlaveMode (TIM_TypeDef * TIMx)</code>                 |
| Function description | Disable the Master/Slave mode.                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                      |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                      |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_TIM_SLAVE_INSTANCE(TIMx) can be used to</li> </ul> |

check whether or not a timer instance can operate as a slave timer.

Reference Manual to  
LL API cross  
reference:

- SMCR MSM LL\_TIM\_DisableMasterSlaveMode

### **LL\_TIM\_IsEnabledMasterSlaveMode**

|                                                   |                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledMasterSlaveMode (TIM_TypeDef * TIMx)</code></b>                                                                        |
| Function description                              | Indicates whether the Master/Slave mode is enabled.                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_SLAVE_INSTANCE(TIMx) can be used to check whether or not a timer instance can operate as a slave timer.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SMCR MSM LL_TIM_IsEnabledMasterSlaveMode</li> </ul>                                                                             |

### **LL\_TIM\_ConfigETR**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_TIM_ConfigETR (TIM_TypeDef * TIMx, uint32_t ETRPolarity, uint32_t ETRPrescaler, uint32_t ETRFilter)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Configure the external trigger (ETR) input.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>ETRPolarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_ETR_POLARITY_NONINVERTED</li> <li>- LL_TIM_ETR_POLARITY_INVERTED</li> </ul> </li> <li>• <b>ETRPrescaler:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_ETR_PRESCALER_DIV1</li> <li>- LL_TIM_ETR_PRESCALER_DIV2</li> <li>- LL_TIM_ETR_PRESCALER_DIV4</li> <li>- LL_TIM_ETR_PRESCALER_DIV8</li> </ul> </li> <li>• <b>ETRFilter:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_ETR_FILTER_FDIV1</li> <li>- LL_TIM_ETR_FILTER_FDIV1_N2</li> <li>- LL_TIM_ETR_FILTER_FDIV1_N4</li> <li>- LL_TIM_ETR_FILTER_FDIV1_N8</li> <li>- LL_TIM_ETR_FILTER_FDIV2_N6</li> <li>- LL_TIM_ETR_FILTER_FDIV2_N8</li> <li>- LL_TIM_ETR_FILTER_FDIV4_N6</li> <li>- LL_TIM_ETR_FILTER_FDIV4_N8</li> <li>- LL_TIM_ETR_FILTER_FDIV8_N6</li> <li>- LL_TIM_ETR_FILTER_FDIV8_N8</li> <li>- LL_TIM_ETR_FILTER_FDIV16_N5</li> </ul> </li> </ul> |

|                                                   |                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_TIM_ETR_FILTER_FDIV16_N6</li> <li>- LL_TIM_ETR_FILTER_FDIV16_N8</li> <li>- LL_TIM_ETR_FILTER_FDIV32_N5</li> <li>- LL_TIM_ETR_FILTER_FDIV32_N6</li> <li>- LL_TIM_ETR_FILTER_FDIV32_N8</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_ETR_INSTANCE(TIMx) can be used to check whether or not a timer instance provides an external trigger input.</li> </ul>                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SMCR ETP LL_TIM_ConfigETR</li> <li>• SMCR ETPS LL_TIM_ConfigETR</li> <li>• SMCR ETF LL_TIM_ConfigETR</li> </ul>                                                                                    |

### LL\_TIM\_EnableBRK

|                                                   |                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableBRK (TIM_TypeDef *<br/>TIMx)</code>                                                                                        |
| Function description                              | Enable the break function.                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a break input.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDTR BKE LL_TIM_EnableBRK</li> </ul>                                                                                      |

### LL\_TIM\_DisableBRK

|                                                   |                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableBRK (TIM_TypeDef *<br/>TIMx)</code>                                                                                       |
| Function description                              | Disable the break function.                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a break input.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDTR BKE LL_TIM_DisableBRK</li> </ul>                                                                                     |

### LL\_TIM\_ConfigBRK

|                      |                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_ConfigBRK (TIM_TypeDef *<br/>TIMx, uint32_t BreakPolarity, uint32_t BreakFilter)</code>                                   |
| Function description | Configure the break input.                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>BreakPolarity:</b> This parameter can be one of the following</li> </ul> |

values:

- LL\_TIM\_BREAK\_POLARITY\_LOW
- LL\_TIM\_BREAK\_POLARITY\_HIGH

- **BreakFilter:** This parameter can be one of the following values:

- LL\_TIM\_BREAK\_FILTER\_FDIV1
- LL\_TIM\_BREAK\_FILTER\_FDIV1\_N2
- LL\_TIM\_BREAK\_FILTER\_FDIV1\_N4
- LL\_TIM\_BREAK\_FILTER\_FDIV1\_N8
- LL\_TIM\_BREAK\_FILTER\_FDIV2\_N6
- LL\_TIM\_BREAK\_FILTER\_FDIV2\_N8
- LL\_TIM\_BREAK\_FILTER\_FDIV4\_N6
- LL\_TIM\_BREAK\_FILTER\_FDIV4\_N8
- LL\_TIM\_BREAK\_FILTER\_FDIV8\_N6
- LL\_TIM\_BREAK\_FILTER\_FDIV8\_N8
- LL\_TIM\_BREAK\_FILTER\_FDIV16\_N5
- LL\_TIM\_BREAK\_FILTER\_FDIV16\_N6
- LL\_TIM\_BREAK\_FILTER\_FDIV16\_N8
- LL\_TIM\_BREAK\_FILTER\_FDIV32\_N5
- LL\_TIM\_BREAK\_FILTER\_FDIV32\_N6
- LL\_TIM\_BREAK\_FILTER\_FDIV32\_N8

#### Return values

- **None**

#### Notes

- Macro IS\_TIM\_BREAK\_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a break input.

#### Reference Manual to LL API cross reference:

- BDTR BKP LL\_TIM\_ConfigBRK
- BDTR BKF LL\_TIM\_ConfigBRK

## LL\_TIM\_EnableBRK2

Function name **`_STATIC_INLINE void LL_TIM_EnableBRK2 (TIM_TypeDef *  
TIMx)`**

#### Function description

Enable the break 2 function.

#### Parameters

- **TIMx:** Timer instance

#### Return values

- **None**

#### Notes

- Macro IS\_TIM\_BKIN2\_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a second break input.

#### Reference Manual to LL API cross reference:

- BDTR BK2E LL\_TIM\_EnableBRK2

## LL\_TIM\_DisableBRK2

Function name **`_STATIC_INLINE void LL_TIM_DisableBRK2 (TIM_TypeDef *  
TIMx)`**

#### Function description

Disable the break 2 function.

#### Parameters

- **TIMx:** Timer instance

---

|                                                   |                                                                                                                                                                       |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"><li>Macro IS_TIM_BKIN2_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a second break input.</li></ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>BDTR BK2E LL_TIM_DisableBRK2</li></ul>                                                                                          |

### LL\_TIM\_ConfigBRK2

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ConfigBRK2 (TIM_TypeDef *<br/>TIMx, uint32_t Break2Polarity, uint32_t Break2Filter)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                              | Configure the break 2 input.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer instance</li> <li><b>Break2Polarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_BREAK2_POLARITY_LOW</li> <li>– LL_TIM_BREAK2_POLARITY_HIGH</li> </ul> </li> <li><b>Break2Filter:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_BREAK2_FILTER_FDIV1</li> <li>– LL_TIM_BREAK2_FILTER_FDIV1_N2</li> <li>– LL_TIM_BREAK2_FILTER_FDIV1_N4</li> <li>– LL_TIM_BREAK2_FILTER_FDIV1_N8</li> <li>– LL_TIM_BREAK2_FILTER_FDIV2_N6</li> <li>– LL_TIM_BREAK2_FILTER_FDIV2_N8</li> <li>– LL_TIM_BREAK2_FILTER_FDIV4_N6</li> <li>– LL_TIM_BREAK2_FILTER_FDIV4_N8</li> <li>– LL_TIM_BREAK2_FILTER_FDIV8_N6</li> <li>– LL_TIM_BREAK2_FILTER_FDIV8_N8</li> <li>– LL_TIM_BREAK2_FILTER_FDIV16_N5</li> <li>– LL_TIM_BREAK2_FILTER_FDIV16_N6</li> <li>– LL_TIM_BREAK2_FILTER_FDIV16_N8</li> <li>– LL_TIM_BREAK2_FILTER_FDIV32_N5</li> <li>– LL_TIM_BREAK2_FILTER_FDIV32_N6</li> <li>– LL_TIM_BREAK2_FILTER_FDIV32_N8</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                                             | <ul style="list-style-type: none"><li>Macro IS_TIM_BKIN2_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a second break input.</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>BDTR BK2P LL_TIM_ConfigBRK2</li><li>BDTR BK2F LL_TIM_ConfigBRK2</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

### LL\_TIM\_SetOffStates

|                      |                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_SetOffStates (TIM_TypeDef *<br/>TIMx, uint32_t OffStateIdle, uint32_t OffStateRun)</code> |
| Function description | Select the outputs off state (enabled v.s.                                                                                  |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>OffStateIdle:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_TIM_OSSI_DISABLE</li> <li>– LL_TIM_OSSI_ENABLE</li> </ul> </li> <li>• <b>OffStateRun:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_TIM_OSSR_DISABLE</li> <li>– LL_TIM_OSSR_ENABLE</li> </ul> </li> </ul> |
| Return values                                     | • <b>None</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a break input.</li> </ul>                                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDTR OSSR LL_TIM_SetOffStates</li> <li>• BDTR OSSR LL_TIM_SetOffStates</li> </ul>                                                                                                                                                                                                                                                                                                                                                             |

### LL\_TIM\_EnableAutomaticOutput

|                                                   |                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_TIM_EnableAutomaticOutput<br/>(TIM_TypeDef * TIMx)</b>                                                                                   |
| Function description                              | Enable automatic output (MOE can be set by software or automatically when a break input is active).                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                    |
| Return values                                     | • <b>None</b>                                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a break input.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDTR AOE LL_TIM_EnableAutomaticOutput</li> </ul>                                                                          |

### LL\_TIM\_DisableAutomaticOutput

|                                                   |                                                                                                                                                                    |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b>_STATIC_INLINE void LL_TIM_DisableAutomaticOutput<br/>(TIM_TypeDef * TIMx)</b>                                                                                  |
| Function description                              | Disable automatic output (MOE can be set only by software).                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                    |
| Return values                                     | • <b>None</b>                                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a break input.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDTR AOE LL_TIM_DisableAutomaticOutput</li> </ul>                                                                         |

**LL\_TIM\_IsEnabledAutomaticOutput**

|                                                   |                                                                                                                                                                                                                        |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_TIM_IsEnabledAutomaticOutput (TIM_TypeDef * TIMx)</code>                                                                                                                         |
| Function description                              | Indicate whether automatic output is enabled.                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                     |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a break input.</li> <li>• BDTR AOE LL_TIM_IsEnabledAutomaticOutput</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: |                                                                                                                                                                                                                        |

**LL\_TIM\_EnableAllOutputs**

|                                                   |                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableAllOutputs<br/>(TIM_TypeDef * TIMx)</code>                                                                                                                                                                                                                              |
| Function description                              | Enable the outputs (set the MOE bit in TIMx_BDTR register).                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>• The MOE bit in TIMx_BDTR register allows to enable /disable the outputs by software and is reset in case of break or break2 event</li> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a break input.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDTR MOE LL_TIM_EnableAllOutputs</li> </ul>                                                                                                                                                                                                                            |

**LL\_TIM\_DisableAllOutputs**

|                                                   |                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableAllOutputs<br/>(TIM_TypeDef * TIMx)</code>                                                                                                                                                                                                                              |
| Function description                              | Disable the outputs (reset the MOE bit in TIMx_BDTR register).                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• The MOE bit in TIMx_BDTR register allows to enable /disable the outputs by software and is reset in case of break or break2 event.</li> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a break input.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BDTR MOE LL_TIM_DisableAllOutputs</li> </ul>                                                                                                                                                                                                                            |

**LL\_TIM\_IsEnabledAllOutputs**

|                                                   |                                                                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledAllOutputs(<br/>    TIM_TypeDef * TIMx)</code>                                                                                                                     |
| Function description                              | Indicates whether outputs are enabled.                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a break input.</li> <li>• BDTR MOE LL_TIM_IsEnabledAllOutputs</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: |                                                                                                                                                                                                                   |

**LL\_TIM\_ConfigDMAburst**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_TIM_ConfigDMAburst(<br/>    TIM_TypeDef * TIMx, uint32_t DMAburstBaseAddress,<br/>    uint32_t DMAburstLength)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Function description | Configures the timer DMA burst feature.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> <li>• <b>DMAburstBaseAddress:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_DMABURST_BASEADDR_CR1</li> <li>- LL_TIM_DMABURST_BASEADDR_CR2</li> <li>- LL_TIM_DMABURST_BASEADDR_SMCR</li> <li>- LL_TIM_DMABURST_BASEADDR_DIER</li> <li>- LL_TIM_DMABURST_BASEADDR_SR</li> <li>- LL_TIM_DMABURST_BASEADDR_EGR</li> <li>- LL_TIM_DMABURST_BASEADDR_CCMR1</li> <li>- LL_TIM_DMABURST_BASEADDR_CCMR2</li> <li>- LL_TIM_DMABURST_BASEADDR_CCER</li> <li>- LL_TIM_DMABURST_BASEADDR_CNT</li> <li>- LL_TIM_DMABURST_BASEADDR_PSC</li> <li>- LL_TIM_DMABURST_BASEADDR_ARR</li> <li>- LL_TIM_DMABURST_BASEADDR_RCR</li> <li>- LL_TIM_DMABURST_BASEADDR_CCR1</li> <li>- LL_TIM_DMABURST_BASEADDR_CCR2</li> <li>- LL_TIM_DMABURST_BASEADDR_CCR3</li> <li>- LL_TIM_DMABURST_BASEADDR_CCR4</li> <li>- LL_TIM_DMABURST_BASEADDR_BDTR</li> <li>- LL_TIM_DMABURST_BASEADDR_CCMR3 (*)</li> <li>- LL_TIM_DMABURST_BASEADDR_CCR5 (*)</li> <li>- LL_TIM_DMABURST_BASEADDR_CCR6 (*) (*) value not defined in all devices</li> <li>- LL_TIM_DMABURST_BASEADDR_OR</li> </ul> </li> <li>• <b>DMAburstLength:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_TIM_DMABURST_LENGTH_1TRANSFER</li> <li>- LL_TIM_DMABURST_LENGTH_2TRANSFERS</li> <li>- LL_TIM_DMABURST_LENGTH_3TRANSFERS</li> </ul> </li> </ul> |

- LL\_TIM\_DMABURST\_LENGTH\_4TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_5TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_6TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_7TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_8TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_9TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_10TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_11TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_12TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_13TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_14TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_15TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_16TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_17TRANSFERS
- LL\_TIM\_DMABURST\_LENGTH\_18TRANSFERS

**Return values**

- **None**

**Notes**

- Macro IS\_TIM\_DMABURST\_INSTANCE(TIMx) can be used to check whether or not a timer instance supports the DMA burst mode.

**Reference Manual to  
LL API cross  
reference:**

- DCR DBL LL\_TIM\_ConfigDMAburst
- DCR DBA LL\_TIM\_ConfigDMAburst

**LL\_TIM\_SetRemap****Function name**

`__STATIC_INLINE void LL_TIM_SetRemap (TIM_TypeDef *  
TIMx, uint32_t Remap)`

**Function description**

Remap TIM inputs (input channel, internal/external triggers).

**Parameters**

- **TIMx:** Timer instance
- **Remap:** Remap params depends on the TIMx. Description available only in CHM version of the User Manual (not in .pdf). Otherwise see Reference Manual description of OR registers.

**Return values**

- **None**

**Notes**

- Macro IS\_TIM\_REMAP\_INSTANCE(TIMx) can be used to check whether or not some timer inputs can be remapped.

**Reference Manual to  
LL API cross  
reference:**

- TIM1\_OR ETR\_RMP LL\_TIM\_SetRemap
- TIM8\_OR ETR\_RMP LL\_TIM\_SetRemap
- TIM20\_OR ETR\_RMP LL\_TIM\_SetRemap
- 

**LL\_TIM\_SetOCRefClearInputSource****Function name**

`__STATIC_INLINE void LL_TIM_SetOCRefClearInputSource  
(TIM_TypeDef * TIMx, uint32_t OCRefClearInputSource)`

**Function description**

Set the OCREF clear input source.

**Parameters**

- **TIMx:** Timer instance
- **OCRefClearInputSource:** This parameter can be one of the

|                                                   |                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | following values:                                                                                                                                                                                                                      |
|                                                   | – LL_TIM_OCREF_CLR_INT_OCREF_CLR                                                                                                                                                                                                       |
|                                                   | – LL_TIM_OCREF_CLR_INT_ETR                                                                                                                                                                                                             |
| Return values                                     | • <b>None</b>                                                                                                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• The OCxREF signal of a given channel can be cleared when a high level is applied on the OCREF_CLR_INPUT</li> <li>• This function can only be used in Output compare and PWM modes.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SMCR OCCS LL_TIM_SetOCRefClearInputSource</li> </ul>                                                                                                                                          |

### LL\_TIM\_ClearFlag\_UPDATE

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_UPDATE(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Clear the update interrupt flag (UIF).                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR UIF LL_TIM_ClearFlag_UPDATE</li> </ul>     |

### LL\_TIM\_IsActiveFlag\_UPDATE

|                                                   |                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_UPDATE(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicate whether update interrupt flag (UIF) is set (update interrupt is pending).            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR UIF LL_TIM_IsActiveFlag_UPDATE</li> </ul>         |

### LL\_TIM\_ClearFlag\_CC1

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_CC1(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Clear the Capture/Compare 1 interrupt flag (CC1F).                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC1IF LL_TIM_ClearFlag_CC1</li> </ul>   |

**LL\_TIM\_IsActiveFlag\_CC1**

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_CC1(<br/>    TIM_TypeDef * TIMx)</code>                |
| Function description                              | Indicate whether Capture/Compare 1 interrupt flag (CC1F) is set (Capture/Compare 1 interrupt is pending). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC1IF LL_TIM_IsActiveFlag_CC1</li> </ul>                      |

**LL\_TIM\_ClearFlag\_CC2**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_CC2 (TIM_TypeDef<br/>    * TIMx)</code> |
| Function description                              | Clear the Capture/Compare 2 interrupt flag (CC2F).                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC2IF LL_TIM_ClearFlag_CC2</li> </ul>   |

**LL\_TIM\_IsActiveFlag\_CC2**

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_CC2(<br/>    TIM_TypeDef * TIMx)</code>                |
| Function description                              | Indicate whether Capture/Compare 2 interrupt flag (CC2F) is set (Capture/Compare 2 interrupt is pending). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC2IF LL_TIM_IsActiveFlag_CC2</li> </ul>                      |

**LL\_TIM\_ClearFlag\_CC3**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_CC3 (TIM_TypeDef<br/>    * TIMx)</code> |
| Function description                              | Clear the Capture/Compare 3 interrupt flag (CC3F).                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC3IF LL_TIM_ClearFlag_CC3</li> </ul>   |

**LL\_TIM\_IsActiveFlag\_CC3**

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_CC3(<br/>    TIM_TypeDef * TIMx)</code>                |
| Function description                              | Indicate whether Capture/Compare 3 interrupt flag (CC3F) is set (Capture/Compare 3 interrupt is pending). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC3IF LL_TIM_IsActiveFlag_CC3</li> </ul>                      |

**LL\_TIM\_ClearFlag\_CC4**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_CC4(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Clear the Capture/Compare 4 interrupt flag (CC4F).                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC4IF LL_TIM_ClearFlag_CC4</li> </ul>   |

**LL\_TIM\_IsActiveFlag\_CC4**

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_CC4(<br/>    TIM_TypeDef * TIMx)</code>                |
| Function description                              | Indicate whether Capture/Compare 4 interrupt flag (CC4F) is set (Capture/Compare 4 interrupt is pending). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC4IF LL_TIM_IsActiveFlag_CC4</li> </ul>                      |

**LL\_TIM\_ClearFlag\_CC5**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_CC5(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Clear the Capture/Compare 5 interrupt flag (CC5F).                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC5IF LL_TIM_ClearFlag_CC5</li> </ul>   |

**LL\_TIM\_IsActiveFlag\_CC5**

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_CC5(<br/>    TIM_TypeDef * TIMx)</code>                |
| Function description                              | Indicate whether Capture/Compare 5 interrupt flag (CC5F) is set (Capture/Compare 5 interrupt is pending). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC5IF LL_TIM_IsActiveFlag_CC5</li> </ul>                      |

**LL\_TIM\_ClearFlag\_CC6**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_CC6 (TIM_TypeDef<br/>    * TIMx)</code> |
| Function description                              | Clear the Capture/Compare 6 interrupt flag (CC6F).                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC6IF LL_TIM_ClearFlag_CC6</li> </ul>   |

**LL\_TIM\_IsActiveFlag\_CC6**

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_CC6(<br/>    TIM_TypeDef * TIMx)</code>                |
| Function description                              | Indicate whether Capture/Compare 6 interrupt flag (CC6F) is set (Capture/Compare 6 interrupt is pending). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC6IF LL_TIM_IsActiveFlag_CC6</li> </ul>                      |

**LL\_TIM\_ClearFlag\_COM**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_COM (TIM_TypeDef<br/>    * TIMx)</code> |
| Function description                              | Clear the commutation interrupt flag (COMIF).                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR COMIF LL_TIM_ClearFlag_COM</li> </ul>   |

**LL\_TIM\_IsActiveFlag\_COM**

Function name `STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_COM(TIM_TypeDef * TIMx)`

Function description Indicate whether commutation interrupt flag (COMIF) is set (commutation interrupt is pending).

Parameters • **TIMx:** Timer instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
SR COMIF LL\_TIM\_IsActiveFlag\_COM

**LL\_TIM\_ClearFlag\_TRIG**

Function name `STATIC_INLINE void LL_TIM_ClearFlag_TRIG(TIM_TypeDef * TIMx)`

Function description Clear the trigger interrupt flag (TIF).

Parameters • **TIMx:** Timer instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
SR TIF LL\_TIM\_ClearFlag\_TRIG

**LL\_TIM\_IsActiveFlag\_TRIG**

Function name `STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_TRIG(TIM_TypeDef * TIMx)`

Function description Indicate whether trigger interrupt flag (TIF) is set (trigger interrupt is pending).

Parameters • **TIMx:** Timer instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
SR TIF LL\_TIM\_IsActiveFlag\_TRIG

**LL\_TIM\_ClearFlag\_BRK**

Function name `STATIC_INLINE void LL_TIM_ClearFlag_BRK(TIM_TypeDef * TIMx)`

Function description Clear the break interrupt flag (BIF).

Parameters • **TIMx:** Timer instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
SR BIF LL\_TIM\_ClearFlag\_BRK

**LL\_TIM\_IsActiveFlag\_BRK**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_BRK(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicate whether break interrupt flag (BIF) is set (break interrupt is pending).           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR BIF LL_TIM_IsActiveFlag_BRK</li> </ul>         |

**LL\_TIM\_ClearFlag\_BRK2**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_BRK2(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Clear the break 2 interrupt flag (B2IF).                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR B2IF LL_TIM_ClearFlag_BRK2</li> </ul>    |

**LL\_TIM\_IsActiveFlag\_BRK2**

|                                                   |                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_BRK2(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicate whether break 2 interrupt flag (B2IF) is set (break 2 interrupt is pending).       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR B2IF LL_TIM_IsActiveFlag_BRK2</li> </ul>        |

**LL\_TIM\_ClearFlag\_CC1OVR**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_CC1OVR(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Clear the Capture/Compare 1 over-capture interrupt flag (CC1OF).                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• SR CC1OF LL_TIM_ClearFlag_CC1OVR</li> </ul>   |

**LL\_TIM\_IsActiveFlag\_CC1OVR**

|                                                   |                                                                                                                         |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_CC1OVR<br/>(TIM_TypeDef * TIMx)</code>                               |
| Function description                              | Indicate whether Capture/Compare 1 over-capture interrupt flag (CC1OF) is set (Capture/Compare 1 interrupt is pending). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                      |
| Reference Manual to<br>LL API cross<br>reference: | SR CC1OF LL_TIM_IsActiveFlag_CC1OVR                                                                                     |

**LL\_TIM\_ClearFlag\_CC2OVR**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_CC2OVR<br/>(TIM_TypeDef * TIMx)</code> |
| Function description                              | Clear the Capture/Compare 2 over-capture interrupt flag (CC2OF).                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | SR CC2OF LL_TIM_ClearFlag_CC2OVR                                                   |

**LL\_TIM\_IsActiveFlag\_CC2OVR**

|                                                   |                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_CC2OVR<br/>(TIM_TypeDef * TIMx)</code>                                            |
| Function description                              | Indicate whether Capture/Compare 2 over-capture interrupt flag (CC2OF) is set (Capture/Compare 2 over-capture interrupt is pending). |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                   |
| Reference Manual to<br>LL API cross<br>reference: | SR CC2OF LL_TIM_IsActiveFlag_CC2OVR                                                                                                  |

**LL\_TIM\_ClearFlag\_CC3OVR**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_ClearFlag_CC3OVR<br/>(TIM_TypeDef * TIMx)</code> |
| Function description                              | Clear the Capture/Compare 3 over-capture interrupt flag (CC3OF).                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | SR CC3OF LL_TIM_ClearFlag_CC3OVR                                                   |

**LL\_TIM\_IsActiveFlag\_CC3OVR**

Function name **`_STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_CC3OVR  
(TIM_TypeDef * TIMx)`**

Function description Indicate whether Capture/Compare 3 over-capture interrupt flag (CC3OF) is set (Capture/Compare 3 over-capture interrupt is pending).

Parameters • **TIMx:** Timer instance

Return values • **State:** of bit (1 or 0).

Reference Manual to SR CC3OF LL\_TIM\_IsActiveFlag\_CC3OVR  
LL API cross reference:

**LL\_TIM\_ClearFlag\_CC4OVR**

Function name **`_STATIC_INLINE void LL_TIM_ClearFlag_CC4OVR  
(TIM_TypeDef * TIMx)`**

Function description Clear the Capture/Compare 4 over-capture interrupt flag (CC4OF).

Parameters • **TIMx:** Timer instance

Return values • **None**

Reference Manual to SR CC4OF LL\_TIM\_ClearFlag\_CC4OVR  
LL API cross reference:

**LL\_TIM\_IsActiveFlag\_CC4OVR**

Function name **`_STATIC_INLINE uint32_t LL_TIM_IsActiveFlag_CC4OVR  
(TIM_TypeDef * TIMx)`**

Function description Indicate whether Capture/Compare 4 over-capture interrupt flag (CC4OF) is set (Capture/Compare 4 over-capture interrupt is pending).

Parameters • **TIMx:** Timer instance

Return values • **State:** of bit (1 or 0).

Reference Manual to SR CC4OF LL\_TIM\_IsActiveFlag\_CC4OVR  
LL API cross reference:

**LL\_TIM\_EnableIT\_UPDATE**

Function name **`_STATIC_INLINE void LL_TIM_EnableIT_UPDATE  
(TIM_TypeDef * TIMx)`**

Function description Enable update interrupt (UIE).

Parameters • **TIMx:** Timer instance

Return values • **None**

Reference Manual to DIER UIE LL\_TIM\_EnableIT\_UPDATE  
LL API cross

reference:

### LL\_TIM\_DisableIT\_UPDATE

Function name **`__STATIC_INLINE void LL_TIM_DisableIT_UPDATE (TIM_TypeDef * TIMx)`**

Function description Disable update interrupt (UIE).

Parameters • **TIMx:** Timer instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
[DIER UIE LL\\_TIM\\_DisableIT\\_UPDATE](#)

### LL\_TIM\_IsEnabledIT\_UPDATE

Function name **`__STATIC_INLINE uint32_t LL_TIM_IsEnabledIT_UPDATE (TIM_TypeDef * TIMx)`**

Function description Indicates whether the update interrupt (UIE) is enabled.

Parameters • **TIMx:** Timer instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
[DIER UIE LL\\_TIM\\_IsEnabledIT\\_UPDATE](#)

### LL\_TIM\_EnableIT\_CC1

Function name **`__STATIC_INLINE void LL_TIM_EnableIT_CC1 (TIM_TypeDef * TIMx)`**

Function description Enable capture/compare 1 interrupt (CC1IE).

Parameters • **TIMx:** Timer instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
[DIER CC1IE LL\\_TIM\\_EnableIT\\_CC1](#)

### LL\_TIM\_DisableIT\_CC1

Function name **`__STATIC_INLINE void LL_TIM_DisableIT_CC1 (TIM_TypeDef * TIMx)`**

Function description Disable capture/compare 1 interrupt (CC1IE).

Parameters • **TIMx:** Timer instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
[DIER CC1IE LL\\_TIM\\_DisableIT\\_CC1](#)

**LL\_TIM\_IsEnabledIT\_CC1**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledIT_CC1(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the capture/compare 1 interrupt (CC1IE) is enabled.                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER CC1IE LL_TIM_IsEnabledIT_CC1</li> </ul>     |

**LL\_TIM\_EnableIT\_CC2**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableIT_CC2 (TIM_TypeDef *<br/>    TIMx)</code> |
| Function description                              | Enable capture/compare 2 interrupt (CC2IE).                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER CC2IE LL_TIM_EnableIT_CC2</li> </ul> |

**LL\_TIM\_DisableIT\_CC2**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableIT_CC2 (TIM_TypeDef *<br/>    TIMx)</code> |
| Function description                              | Disable capture/compare 2 interrupt (CC2IE).                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER CC2IE LL_TIM_DisableIT_CC2</li> </ul> |

**LL\_TIM\_IsEnabledIT\_CC2**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledIT_CC2(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the capture/compare 2 interrupt (CC2IE) is enabled.                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER CC2IE LL_TIM_IsEnabledIT_CC2</li> </ul>     |

**LL\_TIM\_EnableIT\_CC3**

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableIT_CC3 (TIM_TypeDef *<br/>TIMx)</code>   |
| Function description                              | Enable capture/compare 3 interrupt (CC3IE).                                      |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>    |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER CC3IE LL_TIM_EnableIT_CC3</li></ul> |

**LL\_TIM\_DisableIT\_CC3**

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableIT_CC3 (TIM_TypeDef *<br/>* TIMx)</code> |
| Function description                              | Disable capture/compare 3 interrupt (CC3IE).                                      |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>     |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER CC3IE LL_TIM_DisableIT_CC3</li></ul> |

**LL\_TIM\_IsEnabledIT\_CC3**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledIT_CC3<br/>(TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the capture/compare 3 interrupt (CC3IE) is enabled.                 |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>         |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER CC3IE LL_TIM_IsEnabledIT_CC3</li></ul>   |

**LL\_TIM\_EnableIT\_CC4**

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableIT_CC4 (TIM_TypeDef *<br/>TIMx)</code>   |
| Function description                              | Enable capture/compare 4 interrupt (CC4IE).                                      |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>    |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER CC4IE LL_TIM_EnableIT_CC4</li></ul> |

**LL\_TIM\_DisableIT\_CC4**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableIT_CC4 (TIM_TypeDef * TIMx)</code>         |
| Function description                              | Disable capture/compare 4 interrupt (CC4IE).                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER CC4IE LL_TIM_DisableIT_CC4</li> </ul> |

**LL\_TIM\_IsEnabledIT\_CC4**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledIT_CC4 (TIM_TypeDef * TIMx)</code>     |
| Function description                              | Indicates whether the capture/compare 4 interrupt (CC4IE) is enabled.                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER CC4IE LL_TIM_IsEnabledIT_CC4</li> </ul> |

**LL\_TIM\_EnableIT\_COM**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableIT_COM (TIM_TypeDef * TIMx)</code>         |
| Function description                              | Enable commutation interrupt (COMIE).                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER COMIE LL_TIM_EnableIT_COM</li> </ul> |

**LL\_TIM\_DisableIT\_COM**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableIT_COM (TIM_TypeDef * TIMx)</code>         |
| Function description                              | Disable commutation interrupt (COMIE).                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER COMIE LL_TIM_DisableIT_COM</li> </ul> |

**LL\_TIM\_IsEnabledIT\_COM**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_TIM_IsEnabledIT_COM<br/>(TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the commutation interrupt (COMIE) is enabled.                      |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>        |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER COMIE LL_TIM_IsEnabledIT_COM</li></ul>  |

**LL\_TIM\_EnableIT\_TRIG**

|                                                   |                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_TIM_EnableIT_TRIG (TIM_TypeDef<br/>* TIMx)</code>  |
| Function description                              | Enable trigger interrupt (TIE).                                                 |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>   |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER TIE LL_TIM_EnableIT_TRIG</li></ul> |

**LL\_TIM\_DisableIT\_TRIG**

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_TIM_DisableIT_TRIG (TIM_TypeDef<br/>* TIMx)</code>  |
| Function description                              | Disable trigger interrupt (TIE).                                                 |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>    |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER TIE LL_TIM_DisableIT_TRIG</li></ul> |

**LL\_TIM\_IsEnabledIT\_TRIG**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_TIM_IsEnabledIT_TRIG<br/>(TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the trigger interrupt (TIE) is enabled.                             |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>         |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER TIE LL_TIM_IsEnabledIT_TRIG</li></ul>    |

**LL\_TIM\_EnableIT\_BRK**

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableIT_BRK (TIM_TypeDef *<br/>TIMx)</code>   |
| Function description                              | Enable break interrupt (BIE).                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                  |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER BIE LL_TIM_EnableIT_BRK</li> </ul> |

**LL\_TIM\_DisableIT\_BRK**

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableIT_BRK (TIM_TypeDef *<br/>* TIMx)</code> |
| Function description                              | Disable break interrupt (BIE).                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER BIE LL_TIM_DisableIT_BRK</li> </ul> |

**LL\_TIM\_IsEnabledIT\_BRK**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledIT_BRK<br/>(TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the break interrupt (BIE) is enabled.                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER BIE LL_TIM_IsEnabledIT_BRK</li> </ul>   |

**LL\_TIM\_EnableDMAReq\_UPDATE**

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableDMAReq_UPDATE<br/>(TIM_TypeDef * TIMx)</code>   |
| Function description                              | Enable update DMA request (UDE).                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER UDE LL_TIM_EnableDMAReq_UPDATE</li> </ul> |

**LL\_TIM\_DisableDMAReq\_UPDATE**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableDMAReq_UPDATE<br/>(TIM_TypeDef * TIMx)</code> |
| Function description                              | Disable update DMA request (UDE).                                                      |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>          |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER UDE LL_TIM_DisableDMAReq_UPDATE</li></ul> |

**LL\_TIM\_IsEnabledDMAReq\_UPDATE**

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_TIM_IsEnabledDMAReq_UPDATE (TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the update DMA request (UDE) is enabled.                                   |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>                |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER UDE LL_TIM_IsEnabledDMAReq_UPDATE</li></ul>     |

**LL\_TIM\_EnableDMAReq\_CC1**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableDMAReq_CC1<br/>(TIM_TypeDef * TIMx)</code>   |
| Function description                              | Enable capture/compare 1 DMA request (CC1DE).                                        |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>        |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER CC1DE LL_TIM_EnableDMAReq_CC1</li></ul> |

**LL\_TIM\_DisableDMAReq\_CC1**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableDMAReq_CC1<br/>(TIM_TypeDef * TIMx)</code>   |
| Function description                              | Disable capture/compare 1 DMA request (CC1DE).                                        |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>         |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER CC1DE LL_TIM_DisableDMAReq_CC1</li></ul> |

**LL\_TIM\_IsEnabledDMAReq\_CC1**

|                                                   |                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledDMAReq_CC1(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the capture/compare 1 DMA request (CC1DE) is enabled.                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER CC1DE LL_TIM_IsEnabledDMAReq_CC1</li> </ul>     |

**LL\_TIM\_EnableDMAReq\_CC2**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableDMAReq_CC2(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Enable capture/compare 2 DMA request (CC2DE).                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER CC2DE LL_TIM_EnableDMAReq_CC2</li> </ul> |

**LL\_TIM\_DisableDMAReq\_CC2**

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableDMAReq_CC2(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Disable capture/compare 2 DMA request (CC2DE).                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER CC2DE LL_TIM_DisableDMAReq_CC2</li> </ul> |

**LL\_TIM\_IsEnabledDMAReq\_CC2**

|                                                   |                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledDMAReq_CC2(<br/>    TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the capture/compare 2 DMA request (CC2DE) is enabled.                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER CC2DE LL_TIM_IsEnabledDMAReq_CC2</li> </ul>     |

**LL\_TIM\_EnableDMAReq\_CC3**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableDMAReq_CC3<br/>(TIM_TypeDef * TIMx)</code>   |
| Function description                              | Enable capture/compare 3 DMA request (CC3DE).                                        |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>        |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER CC3DE LL_TIM_EnableDMAReq_CC3</li></ul> |

**LL\_TIM\_DisableDMAReq\_CC3**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_DisableDMAReq_CC3<br/>(TIM_TypeDef * TIMx)</code>   |
| Function description                              | Disable capture/compare 3 DMA request (CC3DE).                                        |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>         |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER CC3DE LL_TIM_DisableDMAReq_CC3</li></ul> |

**LL\_TIM\_IsEnabledDMAReq\_CC3**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_TIM_IsEnabledDMAReq_CC3<br/>(TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the capture/compare 3 DMA request (CC3DE) is enabled.                   |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER CC3DE LL_TIM_IsEnabledDMAReq_CC3</li></ul>   |

**LL\_TIM\_EnableDMAReq\_CC4**

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_TIM_EnableDMAReq_CC4<br/>(TIM_TypeDef * TIMx)</code>   |
| Function description                              | Enable capture/compare 4 DMA request (CC4DE).                                        |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>        |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• DIER CC4DE LL_TIM_EnableDMAReq_CC4</li></ul> |

**LL\_TIM\_DisableDMAReq\_CC4**

Function name      **`_STATIC_INLINE void LL_TIM_DisableDMAReq_CC4  
(TIM_TypeDef * TIMx)`**

Function description      Disable capture/compare 4 DMA request (CC4DE).

Parameters      • **TIMx:** Timer instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• DIER CC4DE LL\_TIM\_DisableDMAReq\_CC4

**LL\_TIM\_IsEnabledDMAReq\_CC4**

Function name      **`_STATIC_INLINE uint32_t LL_TIM_IsEnabledDMAReq_CC4  
(TIM_TypeDef * TIMx)`**

Function description      Indicates whether the capture/compare 4 DMA request (CC4DE) is enabled.

Parameters      • **TIMx:** Timer instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
• DIER CC4DE LL\_TIM\_IsEnabledDMAReq\_CC4

**LL\_TIM\_EnableDMAReq\_COM**

Function name      **`_STATIC_INLINE void LL_TIM_EnableDMAReq_COM  
(TIM_TypeDef * TIMx)`**

Function description      Enable commutation DMA request (COMDE).

Parameters      • **TIMx:** Timer instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• DIER COMDE LL\_TIM\_EnableDMAReq\_COM

**LL\_TIM\_DisableDMAReq\_COM**

Function name      **`_STATIC_INLINE void LL_TIM_DisableDMAReq_COM  
(TIM_TypeDef * TIMx)`**

Function description      Disable commutation DMA request (COMDE).

Parameters      • **TIMx:** Timer instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• DIER COMDE LL\_TIM\_DisableDMAReq\_COM

**LL\_TIM\_IsEnabledDMAReq\_COM**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_TIM_IsEnabledDMAReq_COM<br/>(TIM_TypeDef * TIMx)</code>  |
| Function description                              | Indicates whether the commutation DMA request (COMDE) is enabled.                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER COMDE LL_TIM_IsEnabledDMAReq_COM</li> </ul> |

**LL\_TIM\_EnableDMAReq\_TRIG**

|                                                   |                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_TIM_EnableDMAReq_TRIG<br/>(TIM_TypeDef * TIMx)</code>    |
| Function description                              | Enable trigger interrupt (TDE).                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER TDE LL_TIM_EnableDMAReq_TRIG</li> </ul> |

**LL\_TIM\_DisableDMAReq\_TRIG**

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_TIM_DisableDMAReq_TRIG<br/>(TIM_TypeDef * TIMx)</code>    |
| Function description                              | Disable trigger interrupt (TDE).                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER TDE LL_TIM_DisableDMAReq_TRIG</li> </ul> |

**LL\_TIM\_IsEnabledDMAReq\_TRIG**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE uint32_t LL_TIM_IsEnabledDMAReq_TRIG<br/>(TIM_TypeDef * TIMx)</code> |
| Function description                              | Indicates whether the trigger interrupt (TDE) is enabled.                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>        |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• DIER TDE LL_TIM_IsEnabledDMAReq_TRIG</li> </ul>  |

**LL\_TIM\_GenerateEvent\_UPDATE**

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_GenerateEvent_UPDATE<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Generate an update event.                                                                    |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>                |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• EGR UG LL_TIM_GenerateEvent_UPDATE</li></ul>         |

**LL\_TIM\_GenerateEvent\_CC1**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_GenerateEvent_CC1<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Generate Capture/Compare 1 event.                                                         |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• EGR CC1G LL_TIM_GenerateEvent_CC1</li></ul>       |

**LL\_TIM\_GenerateEvent\_CC2**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_GenerateEvent_CC2<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Generate Capture/Compare 2 event.                                                         |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• EGR CC2G LL_TIM_GenerateEvent_CC2</li></ul>       |

**LL\_TIM\_GenerateEvent\_CC3**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_GenerateEvent_CC3<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Generate Capture/Compare 3 event.                                                         |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• EGR CC3G LL_TIM_GenerateEvent_CC3</li></ul>       |

**LL\_TIM\_GenerateEvent\_CC4**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_GenerateEvent_CC4<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Generate Capture/Compare 4 event.                                                         |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• EGR CC4G LL_TIM_GenerateEvent_CC4</li></ul>       |

**LL\_TIM\_GenerateEvent\_COM**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_GenerateEvent_COM<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Generate commutation event.                                                               |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• EGR COMG LL_TIM_GenerateEvent_COM</li></ul>       |

**LL\_TIM\_GenerateEvent\_TRIG**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_GenerateEvent_TRIG<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Generate trigger event.                                                                    |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>              |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• EGR TG LL_TIM_GenerateEvent_TRIG</li></ul>         |

**LL\_TIM\_GenerateEvent\_BRK**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_TIM_GenerateEvent_BRK<br/>(TIM_TypeDef * TIMx)</code></b> |
| Function description                              | Generate break event.                                                                     |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>TIMx:</b> Timer instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• EGR BG LL_TIM_GenerateEvent_BRK</li></ul>         |

**LL\_TIM\_GenerateEvent\_BRK2**

|                                                   |                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_TIM_GenerateEvent_BRK2<br/>(TIM_TypeDef * TIMx)</code> |
| Function description                              | Generate break 2 event.                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                     |
| Reference Manual to<br>LL API cross<br>reference: | • EGR B2G LL_TIM_GenerateEvent_BRK2                                                 |

**LL\_TIM\_DeInit**

|                      |                                                                                                                                                                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_TIM_DeInit (TIM_TypeDef * TIMx)</code>                                                                                                                                                                                     |
| Function description | Set TIMx registers to their reset values.                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer instance</li> </ul>                                                                                                                                                                 |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: TIMx registers are de-initialized</li> <li>– ERROR: invalid TIMx instance</li> </ul> </li> </ul> |

**LL\_TIM\_StructInit**

|                      |                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>void LL_TIM_StructInit (LL_TIM_InitTypeDef * TIM_InitStruct)</code>                                                                                          |
| Function description | Set the fields of the time base unit configuration data structure to their default values.                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIM_InitStruct:</b> pointer to a LL_TIM_InitTypeDef structure (time base unit configuration data structure)</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                    |

**LL\_TIM\_Init**

|                      |                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>ErrorStatus LL_TIM_Init (TIM_TypeDef * TIMx,<br/>LL_TIM_InitTypeDef * TIM_InitStruct)</code>                                                                                                                                       |
| Function description | Configure the TIMx time base unit.                                                                                                                                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer Instance</li> <li>• <b>TIM_InitStruct:</b> pointer to a LL_TIM_InitTypeDef structure (TIMx time base unit configuration data structure)</li> </ul>                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: TIMx registers are de-initialized</li> <li>– ERROR: not applicable</li> </ul> </li> </ul> |

**LL\_TIM\_OC\_StructInit**

|                      |                                                                                                 |
|----------------------|-------------------------------------------------------------------------------------------------|
| Function name        | <code>void LL_TIM_OC_StructInit (LL_TIM_OC_InitTypeDef *<br/>TIM_OC_InitStruct)</code>          |
| Function description | Set the fields of the TIMx output channel configuration data structure to their default values. |

|               |                                                                                                                                                                              |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters    | <ul style="list-style-type: none"> <li>• <b>TIM_OC_InitStruct:</b> pointer to a LL_TIM_OC_InitTypeDef structure (the output channel configuration data structure)</li> </ul> |
| Return values | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                              |

**LL\_TIM\_OC\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_TIM_OC_Init (TIM_TypeDef * TIMx, uint32_t Channel, LL_TIM_OC_InitTypeDef * TIM_OC_InitStruct)</b>                                                                                                                                                                                                                                                                                                                                                                                                              |
| Function description | Configure the TIMx output channel.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> <li>– LL_TIM_CHANNEL_CH5</li> <li>– LL_TIM_CHANNEL_CH6</li> </ul> </li> <li>• <b>TIM_OC_InitStruct:</b> pointer to a LL_TIM_OC_InitTypeDef structure (TIMx output channel configuration data structure)</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: TIMx output channel is initialized</li> <li>– ERROR: TIMx output channel is not initialized</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• OC5 and OC6 are not available for all F3 devices</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                             |

**LL\_TIM\_IC\_StructInit**

|                      |                                                                                                                                                                            |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_TIM_IC_StructInit (LL_TIM_IC_InitTypeDef * TIM_ICInitStruct)</b>                                                                                                |
| Function description | Set the fields of the TIMx input channel configuration data structure to their default values.                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIM_ICInitStruct:</b> pointer to a LL_TIM_IC_InitTypeDef structure (the input channel configuration data structure)</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                            |

**LL\_TIM\_IC\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_TIM_IC_Init (TIM_TypeDef * TIMx, uint32_t Channel, LL_TIM_IC_InitTypeDef * TIM_IC_InitStruct)</b>                                                                                                                                                                                                                                                                                                                                                 |
| Function description | Configure the TIMx input channel.                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer Instance</li> <li>• <b>Channel:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_TIM_CHANNEL_CH1</li> <li>– LL_TIM_CHANNEL_CH2</li> <li>– LL_TIM_CHANNEL_CH3</li> <li>– LL_TIM_CHANNEL_CH4</li> </ul> </li> <li>• <b>TIM_IC_InitStruct:</b> pointer to a LL_TIM_IC_InitTypeDef structure (TIMx input channel configuration data structure)</li> </ul> |

|               |                                                                                                                                                                                                                                                             |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>SUCCESS: TIMx output channel is initialized</li> <li>ERROR: TIMx output channel is not initialized</li> </ul> </li> </ul> |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### LL\_TIM\_ENCODER\_StructInit

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_TIM_ENCODER_StructInit<br/>(LL_TIM_ENCODER_InitTypeDef * TIM_EncoderInitStruct)</b>                                                                                     |
| Function description | Fills each TIM_EncoderInitStruct field with its default value.                                                                                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>TIM_EncoderInitStruct:</b> pointer to a LL_TIM_ENCODER_InitTypeDef structure (encoder interface configuration data structure)</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                      |

### LL\_TIM\_ENCODER\_Init

|                      |                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_TIM_ENCODER_Init (TIM_TypeDef * TIMx,<br/>LL_TIM_ENCODER_InitTypeDef * TIM_EncoderInitStruct)</b>                                                                                                                |
| Function description | Configure the encoder interface of the timer instance.                                                                                                                                                                             |
| Parameters           | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer Instance</li> <li><b>TIM_EncoderInitStruct:</b> pointer to a LL_TIM_ENCODER_InitTypeDef structure (TIMx encoder interface configuration data structure)</li> </ul>       |
| Return values        | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>SUCCESS: TIMx registers are de-initialized</li> <li>ERROR: not applicable</li> </ul> </li> </ul> |

### LL\_TIM\_HALLSENSOR\_StructInit

|                      |                                                                                                                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_TIM_HALLSENSOR_StructInit<br/>(LL_TIM_HALLSENSOR_InitTypeDef *<br/>TIM_HallSensorInitStruct)</b>                                                                                  |
| Function description | Set the fields of the TIMx Hall sensor interface configuration data structure to their default values.                                                                                       |
| Parameters           | <ul style="list-style-type: none"> <li><b>TIM_HallSensorInitStruct:</b> pointer to a LL_TIM_HALLSENSOR_InitTypeDef structure (HALL sensor interface configuration data structure)</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                |

### LL\_TIM\_HALLSENSOR\_Init

|                      |                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_TIM_HALLSENSOR_Init (TIM_TypeDef * TIMx,<br/>LL_TIM_HALLSENSOR_InitTypeDef *<br/>TIM_HallSensorInitStruct)</b>                                                        |
| Function description | Configure the Hall sensor interface of the timer instance.                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>TIMx:</b> Timer Instance</li> <li><b>TIM_HallSensorInitStruct:</b> pointer to a LL_TIM_HALLSENSOR_InitTypeDef structure (TIMx HALL</li> </ul> |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | sensor interface configuration data structure)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Return values | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: TIMx registers are de-initialized</li> <li>– ERROR: not applicable</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Notes         | <ul style="list-style-type: none"> <li>• TIMx CH1, CH2 and CH3 inputs connected through a XOR to the TI1 input channel</li> <li>• TIMx slave mode controller is configured in reset mode. Selected internal trigger is TI1F_ED.</li> <li>• Channel 1 is configured as input, IC1 is mapped on TRC.</li> <li>• Captured value stored in TIMx_CCR1 correspond to the time elapsed between 2 changes on the inputs. It gives information about motor speed.</li> <li>• Channel 2 is configured in output PWM 2 mode.</li> <li>• Compare value stored in TIMx_CCR2 corresponds to the commutation delay.</li> <li>• OC2REF is selected as trigger output on TRGO.</li> <li>• LL_TIM_IC_POLARITY_BOTHEDGE must not be used for TI1 when it is used when TIMx operates in Hall sensor interface mode.</li> </ul> |

### LL\_TIM\_BDTR\_StructInit

|                      |                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_TIM_BDTR_StructInit (LL_TIM_BDTR_InitTypeDef *<br/>TIM_BDTRInitStruct)</b>                                                                                            |
| Function description | Set the fields of the Break and Dead Time configuration data structure to their default values.                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIM_BDTRInitStruct:</b> pointer to a LL_TIM_BDTR_InitTypeDef structure (Break and Dead Time configuration data structure)</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                  |

### LL\_TIM\_BDTR\_Init

|                      |                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_TIM_BDTR_Init (TIM_TypeDef * TIMx,<br/>LL_TIM_BDTR_InitTypeDef * TIM_BDTRInitStruct)</b>                                                                                                                                                                                                                                                         |
| Function description | Configure the Break and Dead Time feature of the timer instance.                                                                                                                                                                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>TIMx:</b> Timer Instance</li> <li>• <b>TIM_BDTRInitStruct:</b> pointer to a LL_TIM_BDTR_InitTypeDef structure(Break and Dead Time configuration data structure)</li> </ul>                                                                                                                                             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:           <ul style="list-style-type: none"> <li>– SUCCESS: Break and Dead Time is initialized</li> <li>– ERROR: not applicable</li> </ul> </li> </ul>                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• As the bits BK2P, BK2E, BK2F[3:0], BKF[3:0], AOE, BKP, BKE, OSS1, OSSR and DTG[7:0] can be write-locked depending on the LOCK configuration, it can be necessary to configure all of them during the first write access to the TIMx_BDTR register.</li> <li>• Macro IS_TIM_BREAK_INSTANCE(TIMx) can be used to</li> </ul> |

- check whether or not a timer instance provides a break input.
- Macro IS\_TIM\_BKIN2\_INSTANCE(TIMx) can be used to check whether or not a timer instance provides a second break input.

## 78.3 TIM Firmware driver defines

### 78.3.1 TIM

#### *Active Input Selection*

LL\_TIM\_ACTIVEINPUT\_DIRECTTI ICx is mapped on TIx

LL\_TIM\_ACTIVEINPUT\_INDIRECTTI ICx is mapped on Tly

LL\_TIM\_ACTIVEINPUT\_TRC ICx is mapped on TRC

#### *Automatic output enable*

LL\_TIM\_AUTOMATICOUTPUT\_DISABLE MOE can be set only by software

LL\_TIM\_AUTOMATICOUTPUT\_ENABLE MOE can be set by software or automatically at the next update event

#### *Break2 Enable*

LL\_TIM\_BREAK2\_DISABLE Break2 function disabled

LL\_TIM\_BREAK2\_ENABLE Break2 function enabled

#### *BREAK2 FILTER*

LL\_TIM\_BREAK2\_FILTER\_FDIV1 No filter, BRK acts asynchronously

LL\_TIM\_BREAK2\_FILTER\_FDIV1\_N2 fSAMPLING=fCK\_INT, N=2

LL\_TIM\_BREAK2\_FILTER\_FDIV1\_N4 fSAMPLING=fCK\_INT, N=4

LL\_TIM\_BREAK2\_FILTER\_FDIV1\_N8 fSAMPLING=fCK\_INT, N=8

LL\_TIM\_BREAK2\_FILTER\_FDIV2\_N6 fSAMPLING=fDTS/2, N=6

LL\_TIM\_BREAK2\_FILTER\_FDIV2\_N8 fSAMPLING=fDTS/2, N=8

LL\_TIM\_BREAK2\_FILTER\_FDIV4\_N6 fSAMPLING=fDTS/4, N=6

LL\_TIM\_BREAK2\_FILTER\_FDIV4\_N8 fSAMPLING=fDTS/4, N=8

LL\_TIM\_BREAK2\_FILTER\_FDIV8\_N6 fSAMPLING=fDTS/8, N=6

LL\_TIM\_BREAK2\_FILTER\_FDIV8\_N8 fSAMPLING=fDTS/8, N=8

LL\_TIM\_BREAK2\_FILTER\_FDIV16\_N5 fSAMPLING=fDTS/16, N=5

LL\_TIM\_BREAK2\_FILTER\_FDIV16\_N6 fSAMPLING=fDTS/16, N=6

LL\_TIM\_BREAK2\_FILTER\_FDIV16\_N8 fSAMPLING=fDTS/16, N=8

LL\_TIM\_BREAK2\_FILTER\_FDIV32\_N5 fSAMPLING=fDTS/32, N=5

LL\_TIM\_BREAK2\_FILTER\_FDIV32\_N6 fSAMPLING=fDTS/32, N=6

LL\_TIM\_BREAK2\_FILTER\_FDIV32\_N8 fSAMPLING=fDTS/32, N=8

#### *BREAK2 POLARITY*

LL\_TIM\_BREAK2\_POLARITY\_LOW Break input BRK2 is active low

`LL_TIM_BREAK2_POLARITY_HIGH` Break input BRK2 is active high

***Break Enable***

`LL_TIM_BREAK_DISABLE` Break function disabled

`LL_TIM_BREAK_ENABLE` Break function enabled

***break filter***

`LL_TIM_BREAK_FILTER_FDIV1` No filter, BRK acts asynchronously

`LL_TIM_BREAK_FILTER_FDIV1_N2` fSAMPLING=fCK\_INT, N=2

`LL_TIM_BREAK_FILTER_FDIV1_N4` fSAMPLING=fCK\_INT, N=4

`LL_TIM_BREAK_FILTER_FDIV1_N8` fSAMPLING=fCK\_INT, N=8

`LL_TIM_BREAK_FILTER_FDIV2_N6` fSAMPLING=fDTS/2, N=6

`LL_TIM_BREAK_FILTER_FDIV2_N8` fSAMPLING=fDTS/2, N=8

`LL_TIM_BREAK_FILTER_FDIV4_N6` fSAMPLING=fDTS/4, N=6

`LL_TIM_BREAK_FILTER_FDIV4_N8` fSAMPLING=fDTS/4, N=8

`LL_TIM_BREAK_FILTER_FDIV8_N6` fSAMPLING=fDTS/8, N=6

`LL_TIM_BREAK_FILTER_FDIV8_N8` fSAMPLING=fDTS/8, N=8

`LL_TIM_BREAK_FILTER_FDIV16_N5` fSAMPLING=fDTS/16, N=5

`LL_TIM_BREAK_FILTER_FDIV16_N6` fSAMPLING=fDTS/16, N=6

`LL_TIM_BREAK_FILTER_FDIV16_N8` fSAMPLING=fDTS/16, N=8

`LL_TIM_BREAK_FILTER_FDIV32_N5` fSAMPLING=fDTS/32, N=5

`LL_TIM_BREAK_FILTER_FDIV32_N6` fSAMPLING=fDTS/32, N=6

`LL_TIM_BREAK_FILTER_FDIV32_N8` fSAMPLING=fDTS/32, N=8

***break polarity***

`LL_TIM_BREAK_POLARITY_LOW` Break input BRK is active low

`LL_TIM_BREAK_POLARITY_HIGH` Break input BRK is active high

***Capture Compare DMA Request***

`LL_TIM_CCDMAREQUEST_CC` CCx DMA request sent when CCx event occurs

`LL_TIM_CCDMAREQUEST_UPDATE` CCx DMA requests sent when update event occurs

***Capture Compare Update Source***

`LL_TIM_CCUPDATESOURCE_COMG_ONLY` Capture/compare control bits are updated by setting the COMG bit only

`LL_TIM_CCUPDATESOURCE_COMG_AND_TRGI` Capture/compare control bits are updated by setting the COMG bit or when a rising edge occurs on trigger input (TRGI)

***Channel***

`LL_TIM_CHANNEL_CH1` Timer input/output channel 1

|                     |                                      |
|---------------------|--------------------------------------|
| LL_TIM_CHANNEL_CH1N | Timer complementary output channel 1 |
| LL_TIM_CHANNEL_CH2  | Timer input/output channel 2         |
| LL_TIM_CHANNEL_CH2N | Timer complementary output channel 2 |
| LL_TIM_CHANNEL_CH3  | Timer input/output channel 3         |
| LL_TIM_CHANNEL_CH3N | Timer complementary output channel 3 |
| LL_TIM_CHANNEL_CH4  | Timer input/output channel 4         |
| LL_TIM_CHANNEL_CH5  | Timer output channel 5               |
| LL_TIM_CHANNEL_CH6  | Timer output channel 6               |

#### **Clock Division**

|                           |                |
|---------------------------|----------------|
| LL_TIM_CLOCKDIVISION_DIV1 | tDTS=tCK_INT   |
| LL_TIM_CLOCKDIVISION_DIV2 | tDTS=2*tCK_INT |
| LL_TIM_CLOCKDIVISION_DIV4 | tDTS=4*tCK_INT |

#### **Clock Source**

|                              |                                                                                 |
|------------------------------|---------------------------------------------------------------------------------|
| LL_TIM_CLOCKSOURCE_INTERNAL  | The timer is clocked by the internal clock provided from the RCC                |
| LL_TIM_CLOCKSOURCE_EXT_MODE1 | Counter counts at each rising or falling edge on a selected input               |
| LL_TIM_CLOCKSOURCE_EXT_MODE2 | Counter counts at each rising or falling edge on the external trigger input ETR |

#### **Counter Direction**

|                              |                           |
|------------------------------|---------------------------|
| LL_TIM_COUNTERDIRECTION_UP   | Timer counter counts up   |
| LL_TIM_COUNTERDIRECTION_DOWN | Timer counter counts down |

#### **Counter Mode**

|                                   |                                                                                                                                                       |
|-----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_TIM_COUNTERMODE_UP             | Counter used as upcounter                                                                                                                             |
| LL_TIM_COUNTERMODE_DOWN           | Counter used as downcounter                                                                                                                           |
| LL_TIM_COUNTERMODE_CENTER_UP      | The counter counts up and down alternatively. Output compare interrupt flags of output channels are set only when the counter is counting down.       |
| LL_TIM_COUNTERMODE_CENTER_DOWN    | The counter counts up and down alternatively. Output compare interrupt flags of output channels are set only when the counter is counting up          |
| LL_TIM_COUNTERMODE_CENTER_UP_DOWN | The counter counts up and down alternatively. Output compare interrupt flags of output channels are set only when the counter is counting up or down. |

#### **DMA Burst Base Address**

|                              |                                                         |
|------------------------------|---------------------------------------------------------|
| LL_TIM_DMABURST_BASEADDR_CR1 | TIMx_CR1 register is the DMA base address for DMA burst |
|------------------------------|---------------------------------------------------------|

|                                |                                                           |
|--------------------------------|-----------------------------------------------------------|
| LL_TIM_DMABURST_BASEADDR_CR2   | TIMx_CR2 register is the DMA base address for DMA burst   |
| LL_TIM_DMABURST_BASEADDR_SMCR  | TIMx_SMCR register is the DMA base address for DMA burst  |
| LL_TIM_DMABURST_BASEADDR_DIER  | TIMx_DIER register is the DMA base address for DMA burst  |
| LL_TIM_DMABURST_BASEADDR_SR    | TIMx_SR register is the DMA base address for DMA burst    |
| LL_TIM_DMABURST_BASEADDR_EGR   | TIMx_EGR register is the DMA base address for DMA burst   |
| LL_TIM_DMABURST_BASEADDR_CCMR1 | TIMx_CCMR1 register is the DMA base address for DMA burst |
| LL_TIM_DMABURST_BASEADDR_CCMR2 | TIMx_CCMR2 register is the DMA base address for DMA burst |
| LL_TIM_DMABURST_BASEADDR_CCER  | TIMx_CCER register is the DMA base address for DMA burst  |
| LL_TIM_DMABURST_BASEADDR_CNT   | TIMx_CNT register is the DMA base address for DMA burst   |
| LL_TIM_DMABURST_BASEADDR_PSC   | TIMx_PSC register is the DMA base address for DMA burst   |
| LL_TIM_DMABURST_BASEADDR_ARR   | TIMx_ARR register is the DMA base address for DMA burst   |
| LL_TIM_DMABURST_BASEADDR_RCR   | TIMx_RCR register is the DMA base address for DMA burst   |
| LL_TIM_DMABURST_BASEADDR_CCR1  | TIMx_CCR1 register is the DMA base address for DMA burst  |
| LL_TIM_DMABURST_BASEADDR_CCR2  | TIMx_CCR2 register is the DMA base address for DMA burst  |
| LL_TIM_DMABURST_BASEADDR_CCR3  | TIMx_CCR3 register is the DMA base address for DMA burst  |
| LL_TIM_DMABURST_BASEADDR_CCR4  | TIMx_CCR4 register is the DMA base address for DMA burst  |
| LL_TIM_DMABURST_BASEADDR_BDTR  | TIMx_BDTR register is the DMA base address for DMA burst  |
| LL_TIM_DMABURST_BASEADDR_CCMR3 | TIMx_CCMR3 register is the DMA base address for DMA burst |
| LL_TIM_DMABURST_BASEADDR_CCR5  | TIMx_CCR5 register is the DMA base address for DMA burst  |
| LL_TIM_DMABURST_BASEADDR_CCR6  | TIMx_CCR6 register is the DMA base address for DMA burst  |
| LL_TIM_DMABURST_BASEADDR_OR    | TIMx_OR register is the DMA base address for DMA burst    |

**DMA Burst Length**

LL\_TIM\_DMABURST\_LENGTH\_1TRANSFER Transfer is done to 1 register starting

|                                    |                                                                           |
|------------------------------------|---------------------------------------------------------------------------|
| LL_TIM_DMABURST_LENGTH_2TRANSFERS  | Transfer is done to 2 registers starting from the DMA burst base address  |
| LL_TIM_DMABURST_LENGTH_3TRANSFERS  | Transfer is done to 3 registers starting from the DMA burst base address  |
| LL_TIM_DMABURST_LENGTH_4TRANSFERS  | Transfer is done to 4 registers starting from the DMA burst base address  |
| LL_TIM_DMABURST_LENGTH_5TRANSFERS  | Transfer is done to 5 registers starting from the DMA burst base address  |
| LL_TIM_DMABURST_LENGTH_6TRANSFERS  | Transfer is done to 6 registers starting from the DMA burst base address  |
| LL_TIM_DMABURST_LENGTH_7TRANSFERS  | Transfer is done to 7 registers starting from the DMA burst base address  |
| LL_TIM_DMABURST_LENGTH_8TRANSFERS  | Transfer is done to 8 registers starting from the DMA burst base address  |
| LL_TIM_DMABURST_LENGTH_9TRANSFERS  | Transfer is done to 9 registers starting from the DMA burst base address  |
| LL_TIM_DMABURST_LENGTH_10TRANSFERS | Transfer is done to 10 registers starting from the DMA burst base address |
| LL_TIM_DMABURST_LENGTH_11TRANSFERS | Transfer is done to 11 registers starting from the DMA burst base address |
| LL_TIM_DMABURST_LENGTH_12TRANSFERS | Transfer is done to 12 registers starting from the DMA burst base address |
| LL_TIM_DMABURST_LENGTH_13TRANSFERS | Transfer is done to 13 registers starting from the DMA burst base address |
| LL_TIM_DMABURST_LENGTH_14TRANSFERS | Transfer is done to 14 registers starting from the DMA burst base address |
| LL_TIM_DMABURST_LENGTH_15TRANSFERS | Transfer is done to 15 registers starting from the DMA burst base address |
| LL_TIM_DMABURST_LENGTH_16TRANSFERS | Transfer is done to 16 registers starting from the DMA burst base address |
| LL_TIM_DMABURST_LENGTH_17TRANSFERS | Transfer is done to 17 registers starting from the DMA burst base address |
| LL_TIM_DMABURST_LENGTH_18TRANSFERS | Transfer is done to 18 registers starting from the DMA burst base address |

**Encoder Mode**

|                            |                                                                                                                     |
|----------------------------|---------------------------------------------------------------------------------------------------------------------|
| LL_TIM_ENCODERMODE_X2_TI1  | Encoder mode 1 - Counter counts up/down on TI2FP2 edge depending on TI1FP1 level                                    |
| LL_TIM_ENCODERMODE_X2_TI2  | Encoder mode 2 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level                                    |
| LL_TIM_ENCODERMODE_X4_TI12 | Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input I |

***External Trigger Filter***

|                             |                                     |
|-----------------------------|-------------------------------------|
| LL_TIM_ETR_FILTER_FDIV1     | No filter, sampling is done at fDTS |
| LL_TIM_ETR_FILTER_FDIV1_N2  | fSAMPLING=fCK_INT, N=2              |
| LL_TIM_ETR_FILTER_FDIV1_N4  | fSAMPLING=fCK_INT, N=4              |
| LL_TIM_ETR_FILTER_FDIV1_N8  | fSAMPLING=fCK_INT, N=8              |
| LL_TIM_ETR_FILTER_FDIV2_N6  | fSAMPLING=fDTS/2, N=6               |
| LL_TIM_ETR_FILTER_FDIV2_N8  | fSAMPLING=fDTS/2, N=8               |
| LL_TIM_ETR_FILTER_FDIV4_N6  | fSAMPLING=fDTS/4, N=6               |
| LL_TIM_ETR_FILTER_FDIV4_N8  | fSAMPLING=fDTS/4, N=8               |
| LL_TIM_ETR_FILTER_FDIV8_N6  | fSAMPLING=fDTS/8, N=8               |
| LL_TIM_ETR_FILTER_FDIV8_N8  | fSAMPLING=fDTS/16, N=5              |
| LL_TIM_ETR_FILTER_FDIV16_N5 | fSAMPLING=fDTS/16, N=6              |
| LL_TIM_ETR_FILTER_FDIV16_N6 | fSAMPLING=fDTS/16, N=8              |
| LL_TIM_ETR_FILTER_FDIV16_N8 | fSAMPLING=fDTS/16, N=5              |
| LL_TIM_ETR_FILTER_FDIV32_N5 | fSAMPLING=fDTS/32, N=5              |
| LL_TIM_ETR_FILTER_FDIV32_N6 | fSAMPLING=fDTS/32, N=6              |
| LL_TIM_ETR_FILTER_FDIV32_N8 | fSAMPLING=fDTS/32, N=8              |

***External Trigger Polarity***

|                                 |                                                          |
|---------------------------------|----------------------------------------------------------|
| LL_TIM_ETR_POLARITY_NONINVERTED | ETR is non-inverted, active at high level or rising edge |
| LL_TIM_ETR_POLARITY_INVERTED    | ETR is inverted, active at low level or falling edge     |

***External Trigger Prescaler***

|                           |                               |
|---------------------------|-------------------------------|
| LL_TIM_ETR_PRESCALER_DIV1 | ETR prescaler OFF             |
| LL_TIM_ETR_PRESCALER_DIV2 | ETR frequency is divided by 2 |
| LL_TIM_ETR_PRESCALER_DIV4 | ETR frequency is divided by 4 |
| LL_TIM_ETR_PRESCALER_DIV8 | ETR frequency is divided by 8 |

***Get Flags Defines***

|                 |                                  |
|-----------------|----------------------------------|
| LL_TIM_SR_UIF   | Update interrupt flag            |
| LL_TIM_SR_CC1IF | Capture/compare 1 interrupt flag |
| LL_TIM_SR_CC2IF | Capture/compare 2 interrupt flag |
| LL_TIM_SR_CC3IF | Capture/compare 3 interrupt flag |
| LL_TIM_SR_CC4IF | Capture/compare 4 interrupt flag |
| LL_TIM_SR_CC5IF | Capture/compare 5 interrupt flag |
| LL_TIM_SR_CC6IF | Capture/compare 6 interrupt flag |
| LL_TIM_SR_COMIF | COM interrupt flag               |
| LL_TIM_SR_TIF   | Trigger interrupt flag           |

---

|                                      |                                                     |
|--------------------------------------|-----------------------------------------------------|
| <code>LL_TIM_SR_BIF</code>           | Break interrupt flag                                |
| <code>LL_TIM_SR_B2IF</code>          | Second break interrupt flag                         |
| <code>LL_TIM_SR_CC1OF</code>         | Capture/Compare 1 overcapture flag                  |
| <code>LL_TIM_SR_CC2OF</code>         | Capture/Compare 2 overcapture flag                  |
| <code>LL_TIM_SR_CC3OF</code>         | Capture/Compare 3 overcapture flag                  |
| <code>LL_TIM_SR_CC4OF</code>         | Capture/Compare 4 overcapture flag                  |
| <b><code>GROUPCH5</code></b>         |                                                     |
| <code>LL_TIM_GROUPCH5_NONE</code>    | No effect of OC5REF on OC1REFC, OC2REFC and OC3REFC |
| <code>LL_TIM_GROUPCH5_OC1REFC</code> | OC1REFC is the logical AND of OC1REFC and OC5REF    |
| <code>LL_TIM_GROUPCH5_OC2REFC</code> | OC2REFC is the logical AND of OC2REFC and OC5REF    |
| <code>LL_TIM_GROUPCH5_OC3REFC</code> | OC3REFC is the logical AND of OC3REFC and OC5REF    |

***Input Configuration Prescaler***

|                                |                                                                                  |
|--------------------------------|----------------------------------------------------------------------------------|
| <code>LL_TIM_ICPSC_DIV1</code> | No prescaler, capture is done each time an edge is detected on the capture input |
| <code>LL_TIM_ICPSC_DIV2</code> | Capture is done once every 2 events                                              |
| <code>LL_TIM_ICPSC_DIV4</code> | Capture is done once every 4 events                                              |
| <code>LL_TIM_ICPSC_DIV8</code> | Capture is done once every 8 events                                              |

***Input Configuration Filter***

|                                         |                                     |
|-----------------------------------------|-------------------------------------|
| <code>LL_TIM_IC_FILTER_FDIV1</code>     | No filter, sampling is done at fDTS |
| <code>LL_TIM_IC_FILTER_FDIV1_N2</code>  | fSAMPLING=fCK_INT, N=2              |
| <code>LL_TIM_IC_FILTER_FDIV1_N4</code>  | fSAMPLING=fCK_INT, N=4              |
| <code>LL_TIM_IC_FILTER_FDIV1_N8</code>  | fSAMPLING=fCK_INT, N=8              |
| <code>LL_TIM_IC_FILTER_FDIV2_N6</code>  | fSAMPLING=fDTS/2, N=6               |
| <code>LL_TIM_IC_FILTER_FDIV2_N8</code>  | fSAMPLING=fDTS/2, N=8               |
| <code>LL_TIM_IC_FILTER_FDIV4_N6</code>  | fSAMPLING=fDTS/4, N=6               |
| <code>LL_TIM_IC_FILTER_FDIV4_N8</code>  | fSAMPLING=fDTS/4, N=8               |
| <code>LL_TIM_IC_FILTER_FDIV8_N6</code>  | fSAMPLING=fDTS/8, N=6               |
| <code>LL_TIM_IC_FILTER_FDIV8_N8</code>  | fSAMPLING=fDTS/8, N=8               |
| <code>LL_TIM_IC_FILTER_FDIV16_N5</code> | fSAMPLING=fDTS/16, N=5              |
| <code>LL_TIM_IC_FILTER_FDIV16_N6</code> | fSAMPLING=fDTS/16, N=6              |
| <code>LL_TIM_IC_FILTER_FDIV16_N8</code> | fSAMPLING=fDTS/16, N=8              |
| <code>LL_TIM_IC_FILTER_FDIV32_N5</code> | fSAMPLING=fDTS/32, N=5              |
| <code>LL_TIM_IC_FILTER_FDIV32_N6</code> | fSAMPLING=fDTS/32, N=6              |
| <code>LL_TIM_IC_FILTER_FDIV32_N8</code> | fSAMPLING=fDTS/32, N=8              |

***Input Configuration Polarity***

|                             |                                                                                          |
|-----------------------------|------------------------------------------------------------------------------------------|
| LL_TIM_IC_POLARITY_RISING   | The circuit is sensitive to TIxFP1 rising edge, TIxFP1 is not inverted                   |
| LL_TIM_IC_POLARITY_FALLING  | The circuit is sensitive to TIxFP1 falling edge, TIxFP1 is inverted                      |
| LL_TIM_IC_POLARITY_BOTHEDGE | The circuit is sensitive to both TIxFP1 rising and falling edges, TIxFP1 is not inverted |

***IT Defines***

|                   |                                    |
|-------------------|------------------------------------|
| LL_TIM_DIER_UIE   | Update interrupt enable            |
| LL_TIM_DIER_CC1IE | Capture/compare 1 interrupt enable |
| LL_TIM_DIER_CC2IE | Capture/compare 2 interrupt enable |
| LL_TIM_DIER_CC3IE | Capture/compare 3 interrupt enable |
| LL_TIM_DIER_CC4IE | Capture/compare 4 interrupt enable |
| LL_TIM_DIER_COMIE | COM interrupt enable               |
| LL_TIM_DIER_TIE   | Trigger interrupt enable           |
| LL_TIM_DIER_BIE   | Break interrupt enable             |

***Lock Level***

|                      |                                      |
|----------------------|--------------------------------------|
| LL_TIM_LOCKLEVEL_OFF | LOCK OFF - No bit is write protected |
| LL_TIM_LOCKLEVEL_1   | LOCK Level 1                         |
| LL_TIM_LOCKLEVEL_2   | LOCK Level 2                         |
| LL_TIM_LOCKLEVEL_3   | LOCK Level 3                         |

***Output Configuration Idle State***

|                         |                                                           |
|-------------------------|-----------------------------------------------------------|
| LL_TIM_OCIDLESTATE_LOW  | OCx=0 (after a dead-time if OC is implemented) when MOE=0 |
| LL_TIM_OCIDLESTATE_HIGH | OCx=1 (after a dead-time if OC is implemented) when MOE=0 |

***Output Configuration Mode***

|                               |                                                                                                                                                                   |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_TIM_OCMODE_FROZEN          | The comparison between the output compare register TIMx_CCRy and the counter TIMx_CNT has no effect on the output channel level                                   |
| LL_TIM_OCMODE_ACTIVE          | OCyREF is forced high on compare match                                                                                                                            |
| LL_TIM_OCMODE_INACTIVE        | OCyREF is forced low on compare match                                                                                                                             |
| LL_TIM_OCMODE_TOGGLE          | OCyREF toggles on compare match                                                                                                                                   |
| LL_TIM_OCMODE_FORCED_INACTIVE | OCyREF is forced low                                                                                                                                              |
| LL_TIM_OCMODE_FORCED_ACTIVE   | OCyREF is forced high                                                                                                                                             |
| LL_TIM_OCMODE_PWM1            | In upcounting, channel y is active as long as TIMx_CNT<TIMx_CCRy else inactive. In downcounting, channel y is inactive as long as TIMx_CNT>TIMx_CCRy else active. |

|                               |                                                                                                                                                                  |
|-------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_TIM_OCMODE_PWM2            | In upcounting, channel y is inactive as long as TIMx_CNT<TIMx_CCRy else active. In downcounting, channel y is active as long as TIMx_CNT>TIMx_CCRy else inactive |
| LL_TIM_OCMODE_RETRIG_OPM1     | Retriggerable OPM mode 1                                                                                                                                         |
| LL_TIM_OCMODE_RETRIG_OPM2     | Retriggerable OPM mode 2                                                                                                                                         |
| LL_TIM_OCMODE_COMBINED_PWM1   | Combined PWM mode 1                                                                                                                                              |
| LL_TIM_OCMODE_COMBINED_PWM2   | Combined PWM mode 2                                                                                                                                              |
| LL_TIM_OCMODE_ASSYMETRIC_PWM1 | Asymmetric PWM mode 1                                                                                                                                            |
| LL_TIM_OCMODE_ASSYMETRIC_PWM2 | Asymmetric PWM mode 2                                                                                                                                            |

#### ***Output Configuration Polarity***

|                        |                 |
|------------------------|-----------------|
| LL_TIM_OCPOLARITY_HIGH | OCx active high |
| LL_TIM_OCPOLARITY_LOW  | OCx active low  |

#### ***OCREF clear input selection***

|                                |                                                   |
|--------------------------------|---------------------------------------------------|
| LL_TIM_OCREF_CLR_INT_OCREF_CLR | OCREF_CLR_INT is connected to the OCREF_CLR input |
| LL_TIM_OCREF_CLR_INT_ETR       | OCREF_CLR_INT is connected to ETRF                |

#### ***Output Configuration State***

|                        |                                                      |
|------------------------|------------------------------------------------------|
| LL_TIM_OCSTATE_DISABLE | OCx is not active                                    |
| LL_TIM_OCSTATE_ENABLE  | OCx signal is output on the corresponding output pin |

#### ***One Pulse Mode***

|                               |                                                 |
|-------------------------------|-------------------------------------------------|
| LL_TIM_ONEPULSEMODE_SINGLE    | Counter is not stopped at update event          |
| LL_TIM_ONEPULSEMODE_REPEATIVE | Counter stops counting at the next update event |

#### ***OSSI***

|                     |                                                                                                                               |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------|
| LL_TIM_OSSI_DISABLE | When inactive, OCx/OCxN outputs are disabled                                                                                  |
| LL_TIM_OSSI_ENABLE  | When inactive, OCx/OCxN outputs are first forced with their inactive level then forced to their idle level after the deadtime |

#### ***OSSR***

|                     |                                                                                                    |
|---------------------|----------------------------------------------------------------------------------------------------|
| LL_TIM_OSSR_DISABLE | When inactive, OCx/OCxN outputs are disabled                                                       |
| LL_TIM_OSSR_ENABLE  | When inactive, OCx/OCxN outputs are enabled with their inactive level as soon as CCxE=1 or CCxNE=1 |

#### ***Slave Mode***

|                           |                                                                                         |
|---------------------------|-----------------------------------------------------------------------------------------|
| LL_TIM_SLAVEMODE_DISABLED | Slave mode disabled                                                                     |
| LL_TIM_SLAVEMODE_RESET    | Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter |
| LL_TIM_SLAVEMODE_GATED    | Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high         |

|                                        |                                                                                                                                                                         |
|----------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LL_TIM_SLAVEMODE_TRIGGER               | Trigger Mode - The counter starts at a rising edge of the trigger TRGI                                                                                                  |
| LL_TIM_SLAVEMODE_COMBINED_RESETTRIGGER | Combined reset + trigger mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter, generates an update of the registers and starts the counter |

***TIM16 External Input Ch1 Remap***

|                             |                                                            |
|-----------------------------|------------------------------------------------------------|
| LL_TIM_TIM16_TI1_RMP_GPIO   | TIM16 input capture 1 is connected to GPIO                 |
| LL_TIM_TIM16_TI1_RMP_RTC    | TIM16 input capture 1 is connected to RTC wakeup interrupt |
| LL_TIM_TIM16_TI1_RMP_HSE_32 | TIM16 input capture 1 is connected to HSE/32 clock         |
| LL_TIM_TIM16_TI1_RMP_MCO    | TIM16 input capture 1 is connected to MCO                  |

***TIM1 External Trigger ADC1 Remap***

|                               |                                                     |
|-------------------------------|-----------------------------------------------------|
| LL_TIM_TIM1_ETR_ADC1_RMP_NC   | TIM1_ETR is not connected to ADC1 analog watchdog x |
| LL_TIM_TIM1_ETR_ADC1_RMP_AWD1 | TIM1_ETR is connected to ADC1 analog watchdog 1     |
| LL_TIM_TIM1_ETR_ADC1_RMP_AWD2 | TIM1_ETR is connected to ADC1 analog watchdog 2     |
| LL_TIM_TIM1_ETR_ADC1_RMP_AWD3 | TIM1_ETR is connected to ADC1 analog watchdog 3     |

***TIM1 External Trigger ADC4 Remap***

|                               |                                                     |
|-------------------------------|-----------------------------------------------------|
| LL_TIM_TIM1_ETR_ADC4_RMP_NC   | TIM1_ETR is not connected to ADC4 analog watchdog x |
| LL_TIM_TIM1_ETR_ADC4_RMP_AWD1 | TIM1_ETR is connected to ADC4 analog watchdog 1     |
| LL_TIM_TIM1_ETR_ADC4_RMP_AWD2 | TIM1_ETR is connected to ADC4 analog watchdog 2     |
| LL_TIM_TIM1_ETR_ADC4_RMP_AWD3 | TIM1_ETR is connected to ADC4 analog watchdog 3     |

***TIM8 External Trigger ADC2 Remap***

|                               |                                                     |
|-------------------------------|-----------------------------------------------------|
| LL_TIM_TIM8_ETR_ADC2_RMP_NC   | TIM8_ETR is not connected to ADC2 analog watchdog x |
| LL_TIM_TIM8_ETR_ADC2_RMP_AWD1 | TIM8_ETR is connected to ADC2 analog watchdog       |
| LL_TIM_TIM8_ETR_ADC2_RMP_AWD2 | TIM8_ETR is connected to ADC2 analog watchdog 2     |
| LL_TIM_TIM8_ETR_ADC2_RMP_AWD3 | TIM8_ETR is connected to ADC2 analog watchdog 3     |

***TIM8 External Trigger ADC3 Remap***

|                               |                                                     |
|-------------------------------|-----------------------------------------------------|
| LL_TIM_TIM8_ETR_ADC3_RMP_NC   | TIM8_ETR is not connected to ADC3 analog watchdog x |
| LL_TIM_TIM8_ETR_ADC3_RMP_AWD1 | TIM8_ETR is connected to ADC3 analog watchdog 1     |
| LL_TIM_TIM8_ETR_ADC3_RMP_AWD2 | TIM8_ETR is connected to ADC3 analog watchdog 2     |
| LL_TIM_TIM8_ETR_ADC3_RMP_AWD3 | TIM8_ETR is connected to ADC3 analog watchdog 3     |

***Trigger Output***

|                    |                                                             |
|--------------------|-------------------------------------------------------------|
| LL_TIM_TRGO_RESET  | UG bit from the TIMx_EGR register is used as trigger output |
| LL_TIM_TRGO_ENABLE | Counter Enable signal (CNT_EN) is used as trigger output    |
| LL_TIM_TRGO_UPDATE | Update event is used as trigger output                      |
| LL_TIM_TRGO_CC1IF  | CC1 capture or a compare match is used as trigger output    |
| LL_TIM_TRGO_OC1REF | OC1REF signal is used as trigger output                     |
| LL_TIM_TRGO_OC2REF | OC2REF signal is used as trigger output                     |
| LL_TIM_TRGO_OC3REF | OC3REF signal is used as trigger output                     |
| LL_TIM_TRGO_OC4REF | OC4REF signal is used as trigger output                     |

***Trigger Output 2***

|                                |                                                               |
|--------------------------------|---------------------------------------------------------------|
| LL_TIM_TRGO2_RESET             | UG bit from the TIMx_EGR register is used as trigger output 2 |
| LL_TIM_TRGO2_ENABLE            | Counter Enable signal (CNT_EN) is used as trigger output 2    |
| LL_TIM_TRGO2_UPDATE            | Update event is used as trigger output 2                      |
| LL_TIM_TRGO2_CC1F              | CC1 capture or a compare match is used as trigger output 2    |
| LL_TIM_TRGO2_OC1               | OC1REF signal is used as trigger output 2                     |
| LL_TIM_TRGO2_OC2               | OC2REF signal is used as trigger output 2                     |
| LL_TIM_TRGO2_OC3               | OC3REF signal is used as trigger output 2                     |
| LL_TIM_TRGO2_OC4               | OC4REF signal is used as trigger output 2                     |
| LL_TIM_TRGO2_OC5               | OC5REF signal is used as trigger output 2                     |
| LL_TIM_TRGO2_OC6               | OC6REF signal is used as trigger output 2                     |
| LL_TIM_TRGO2_OC4_RISINGFALLING | OC4REF rising or falling edges are used as trigger output 2   |
| LL_TIM_TRGO2_OC6_RISINGFALLING | OC6REF rising or falling edges are used                       |

|                                     |                                                                    |
|-------------------------------------|--------------------------------------------------------------------|
|                                     | as trigger output 2                                                |
| LL_TIM_TRGO2_OC4_RISING_OC6_RISING  | OC4REF or OC6REF rising edges are used as trigger output 2         |
| LL_TIM_TRGO2_OC4_RISING_OC6_FALLING | OC4REF rising or OC6REF falling edges are used as trigger output 2 |
| LL_TIM_TRGO2_OC5_RISING_OC6_RISING  | OC5REF or OC6REF rising edges are used as trigger output 2         |
| LL_TIM_TRGO2_OC5_RISING_OC6_FALLING | OC5REF rising or OC6REF falling edges are used as trigger output 2 |

**Trigger Selection**

|                   |                                                           |
|-------------------|-----------------------------------------------------------|
| LL_TIM_TS_ITR0    | Internal Trigger 0 (ITR0) is used as trigger input        |
| LL_TIM_TS_ITR1    | Internal Trigger 1 (ITR1) is used as trigger input        |
| LL_TIM_TS_ITR2    | Internal Trigger 2 (ITR2) is used as trigger input        |
| LL_TIM_TS_ITR3    | Internal Trigger 3 (ITR3) is used as trigger input        |
| LL_TIM_TS_TI1F_ED | TI1 Edge Detector (TI1F_ED) is used as trigger input      |
| LL_TIM_TS_TI1FP1  | Filtered Timer Input 1 (TI1FP1) is used as trigger input  |
| LL_TIM_TS_TI2FP2  | Filtered Timer Input 2 (TI12P2) is used as trigger input  |
| LL_TIM_TS_ETRF    | Filtered external Trigger (ETRF) is used as trigger input |

**Update Source**

|                             |                                                                                                                                   |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| LL_TIM_UPDATESOURCE_REGULAR | Counter overflow/underflow, Setting the UG bit or Update generation through the slave mode controller generates an update request |
| LL_TIM_UPDATESOURCE_COUNTER | Only counter overflow/underflow generates an update request                                                                       |

**Exported Macros**

|                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>__LL_TIM_GETFLAG_UIFCPY</u> | <b>Description:</b><br><br><ul style="list-style-type: none"> <li>• HELPER macro retrieving the UIFCPY flag from the counter value.</li> </ul> <b>Parameters:</b><br><ul style="list-style-type: none"> <li>• <u>__CNT__</u>: Counter value</li> </ul> <b>Return value:</b><br><ul style="list-style-type: none"> <li>• UIF: status bit</li> </ul> <b>Notes:</b><br><ul style="list-style-type: none"> <li>• ex: __LL_TIM_GETFLAG_UIFCPY<br/>(LL_TIM_GetCounter()); Relevant only if UIF flag remapping has been enabled (UIF status bit is copied to TIMx_CNT register bit 31)</li> </ul> |
| <u>__LL_TIM_CALC_DEADTIME</u>  | <b>Description:</b><br><br><ul style="list-style-type: none"> <li>• HELPER macro calculating DTG[0:7] in the TIMx_BDTR register to achieve the requested dead time duration.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                    |

**Parameters:**

- `__TIMCLK__`: timer input clock frequency (in Hz)
- `__CKD__`: This parameter can be one of the following values:
  - `LL_TIM_CLOCKDIVISION_DIV1`
  - `LL_TIM_CLOCKDIVISION_DIV2`
  - `LL_TIM_CLOCKDIVISION_DIV4`
- `__DT__`: deadtime duration (in ns)

**Return value:**

- `DTG[0:7]`

**Notes:**

- ex: `__LL_TIM_CALC_DEADTIME (80000000, LL_TIM_GetClockDivision (), 120);`

[\\_\\_LL\\_TIM\\_CALC\\_PSC](#)**Description:**

- `LL_HELPER` macro calculating the prescaler value to achieve the required counter clock frequency.

**Parameters:**

- `__TIMCLK__`: timer input clock frequency (in Hz)
- `__CNTCLK__`: counter clock frequency (in Hz)

**Return value:**

- Prescaler: value (between `Min_Data=0` and `Max_Data=65535`)

**Notes:**

- ex: `__LL_TIM_CALC_PSC (80000000, 1000000);`

[\\_\\_LL\\_TIM\\_CALC\\_ARR](#)**Description:**

- `LL_HELPER` macro calculating the auto-reload value to achieve the required output signal frequency.

**Parameters:**

- `__TIMCLK__`: timer input clock frequency (in Hz)
- `__PSC__`: prescaler
- `__FREQ__`: output signal frequency (in Hz)

**Return value:**

- Auto-reload: value (between `Min_Data=0` and `Max_Data=65535`)

**Notes:**

- ex: `__LL_TIM_CALC_ARR (1000000, LL_TIM_GetPrescaler (), 10000);`

[\\_\\_LL\\_TIM\\_CALC\\_DELAY](#)**Description:**

- `LL_HELPER` macro calculating the compare value required to achieve the required timer output compare active/inactive delay.

**Parameters:**

- `__TIMCLK__`: timer input clock frequency (in Hz)
- `__PSC__`: prescaler
- `__DELAY__`: timer output compare active/inactive delay (in us)

**Return value:**

- Compare: value (between Min\_Data=0 and Max\_Data=65535)

**Notes:**

- ex: `__LL_TIM_CALC_DELAY (1000000, LL_TIM_GetPrescaler (), 10);`

**\_\_LL\_TIM\_CALC\_PULSE****Description:**

- HELPER macro calculating the auto-reload value to achieve the required pulse duration (when the timer operates in one pulse mode).

**Parameters:**

- `__TIMCLK__`: timer input clock frequency (in Hz)
- `__PSC__`: prescaler
- `__DELAY__`: timer output compare active/inactive delay (in us)
- `__PULSE__`: pulse duration (in us)

**Return value:**

- Auto-reload: value (between Min\_Data=0 and Max\_Data=65535)

**Notes:**

- ex: `__LL_TIM_CALC_PULSE (1000000, LL_TIM_GetPrescaler (), 10, 20);`

**\_\_LL\_TIM\_GET\_ICPSC\_RATIO****Description:**

- HELPER macro retrieving the ratio of the input capture prescaler.

**Parameters:**

- `__ICPSC__`: This parameter can be one of the following values:
  - `LL_TIM_ICPSC_DIV1`
  - `LL_TIM_ICPSC_DIV2`
  - `LL_TIM_ICPSC_DIV4`
  - `LL_TIM_ICPSC_DIV8`

**Return value:**

- Input: capture prescaler ratio (1, 2, 4 or 8)

**Notes:**

- ex: `__LL_TIM_GET_ICPSC_RATIO (LL_TIM_IC_GetPrescaler ());`

***Common Write and read registers Macros*****LL\_TIM\_WriteReg****Description:**

- Write a value in TIM register.

**Parameters:**

- \_\_INSTANCE\_\_: TIM Instance
- \_\_REG\_\_: Register to be written
- \_\_VALUE\_\_: Value to be written in the register

**Return value:**

- None

**LL\_TIM\_ReadReg****Description:**

- Read a value in TIM register.

**Parameters:**

- \_\_INSTANCE\_\_: TIM Instance
- \_\_REG\_\_: Register to be read

**Return value:**

- Register: value

## 79 LL USART Generic Driver

### 79.1 USART Firmware driver registers structures

#### 79.1.1 LL\_USART\_InitTypeDef

##### Data Fields

- *uint32\_t BaudRate*
- *uint32\_t DataWidth*
- *uint32\_t StopBits*
- *uint32\_t Parity*
- *uint32\_t TransferDirection*
- *uint32\_t HardwareFlowControl*
- *uint32\_t OverSampling*

##### Field Documentation

- ***uint32\_t LL\_USART\_InitTypeDef::BaudRate***  
This field defines expected Usart communication baud rate. This feature can be modified afterwards using unitary function **LL\_USART\_SetBaudRate()**.
- ***uint32\_t LL\_USART\_InitTypeDef::DataWidth***  
Specifies the number of data bits transmitted or received in a frame. This parameter can be a value of **USART\_LL\_EC\_DATAWIDTH**. This feature can be modified afterwards using unitary function **LL\_USART\_SetDataWidth()**.
- ***uint32\_t LL\_USART\_InitTypeDef::StopBits***  
Specifies the number of stop bits transmitted. This parameter can be a value of **USART\_LL\_EC\_STOPBITS**. This feature can be modified afterwards using unitary function **LL\_USART\_SetStopBitsLength()**.
- ***uint32\_t LL\_USART\_InitTypeDef::Parity***  
Specifies the parity mode. This parameter can be a value of **USART\_LL\_EC\_PARITY**. This feature can be modified afterwards using unitary function **LL\_USART\_SetParity()**.
- ***uint32\_t LL\_USART\_InitTypeDef::TransferDirection***  
Specifies whether the Receive and/or Transmit mode is enabled or disabled. This parameter can be a value of **USART\_LL\_EC\_DIRECTION**. This feature can be modified afterwards using unitary function **LL\_USART\_SetTransferDirection()**.
- ***uint32\_t LL\_USART\_InitTypeDef::HardwareFlowControl***  
Specifies whether the hardware flow control mode is enabled or disabled. This parameter can be a value of **USART\_LL\_EC\_HWCONTROL**. This feature can be modified afterwards using unitary function **LL\_USART\_SetHWFlowCtrl()**.
- ***uint32\_t LL\_USART\_InitTypeDef::OverSampling***  
Specifies whether USART oversampling mode is 16 or 8. This parameter can be a value of **USART\_LL\_EC\_OVERSAMPLING**. This feature can be modified afterwards using unitary function **LL\_USART\_SetOverSampling()**.

#### 79.1.2 LL\_USART\_ClockInitTypeDef

##### Data Fields

- *uint32\_t ClockOutput*
- *uint32\_t ClockPolarity*
- *uint32\_t ClockPhase*
- *uint32\_t LastBitClockPulse*

### Field Documentation

- ***uint32\_t LL\_USART\_ClockInitTypeDef::ClockOutput***  
Specifies whether the USART clock is enabled or disabled. This parameter can be a value of **USART\_LL\_EC\_CLOCK**.USART HW configuration can be modified afterwards using unitary functions **LL\_USART\_EnableSCLKOutput()** or **LL\_USART\_DisableSCLKOutput()**. For more details, refer to description of this function.
- ***uint32\_t LL\_USART\_ClockInitTypeDef::ClockPolarity***  
Specifies the steady state of the serial clock. This parameter can be a value of **USART\_LL\_EC\_POLARITY**.USART HW configuration can be modified afterwards using unitary functions **LL\_USART\_SetClockPolarity()**. For more details, refer to description of this function.
- ***uint32\_t LL\_USART\_ClockInitTypeDef::ClockPhase***  
Specifies the clock transition on which the bit capture is made. This parameter can be a value of **USART\_LL\_EC\_PHASE**.USART HW configuration can be modified afterwards using unitary functions **LL\_USART\_SetClockPhase()**. For more details, refer to description of this function.
- ***uint32\_t LL\_USART\_ClockInitTypeDef::LastBitClockPulse***  
Specifies whether the clock pulse corresponding to the last transmitted data bit (MSB) has to be output on the SCLK pin in synchronous mode. This parameter can be a value of **USART\_LL\_EC\_LASTCLKPULSE**.USART HW configuration can be modified afterwards using unitary functions **LL\_USART\_SetLastClkPulseOutput()**. For more details, refer to description of this function.

## 79.2 USART Firmware driver API description

### 79.2.1 Detailed description of functions

#### LL\_USART\_Enable

|                                                   |                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_Enable (USART_TypeDef * USARTx)</code>        |
| Function description                              | USART Enable.                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 UE LL_USART_Enable</li> </ul>        |

#### LL\_USART\_Disable

|                      |                                                                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_USART_Disable (USART_TypeDef * USARTx)</code>                                                                                                                                                        |
| Function description | USART Disable (all USART prescalers and outputs are disabled)                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• When USART is disabled, USART prescalers and outputs are stopped immediately, and current operations are discarded. The configuration of the USART is kept, but all the status</li> </ul> |

flags, in the USARTx\_ISR are set to their default values.

Reference Manual to  
LL API cross  
reference:

- CR1 UE LL\_USART\_Disable

### **LL\_USART\_IsEnabled**

Function name **\_STATIC\_INLINE uint32\_t LL\_USART\_IsEnabled(**USART\_TypeDef \* USARTx**)**

Function description Indicate if USART is enabled.

Parameters • **USARTx**: USART Instance

Return values • **State**: of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
• CR1 UE LL\_USART\_IsEnabled

### **LL\_USART\_EnableInStopMode**

Function name **\_STATIC\_INLINE void LL\_USART\_EnableInStopMode(**USART\_TypeDef \* USARTx**)**

Function description USART enabled in STOP Mode.

Parameters • **USARTx**: USART Instance

Return values • **None**

Notes

- When this function is enabled, USART is able to wake up the MCU from Stop mode, provided that USART clock selection is HSI or LSE in RCC.
- Macro IS\_UART\_WAKEUP\_FROMSTOP\_INSTANCE(USARTx) can be used to check whether or not Wake-up from Stop mode feature is supported by the USARTx instance.

Reference Manual to  
LL API cross  
reference:  
• CR1 UESM LL\_USART\_EnableInStopMode

### **LL\_USART\_DisableInStopMode**

Function name **\_STATIC\_INLINE void LL\_USART\_DisableInStopMode(**USART\_TypeDef \* USARTx**)**

Function description USART disabled in STOP Mode.

Parameters • **USARTx**: USART Instance

Return values • **None**

Notes

- When this function is disabled, USART is not able to wake up the MCU from Stop mode
- Macro IS\_UART\_WAKEUP\_FROMSTOP\_INSTANCE(USARTx) can be used to check whether or not Wake-up from Stop mode feature is supported by the USARTx instance.

- Reference Manual to LL API cross reference:
- CR1 UESM LL\_USART\_DisableInStopMode

### **LL\_USART\_IsEnabledInStopMode**

|                                             |                                                                                                                                                                                                                         |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_USART_IsEnabledInStopMode(USART_TypeDef * USARTx)</code></b>                                                                                                                        |
| Function description                        | Indicate if USART is enabled in STOP Mode (able to wake up MCU from Stop mode or not)                                                                                                                                   |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                       |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                      |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro <code>IS_UART_WAKEUP_FROMSTOP_INSTANCE(USARTx)</code> can be used to check whether or not Wake-up from Stop mode feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR1 UESM LL_USART_IsEnabledInStopMode</li> </ul>                                                                                                                               |

### **LL\_USART\_EnableDirectionRx**

|                                             |                                                                                            |
|---------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_USART_EnableDirectionRx(USART_TypeDef * USARTx)</code></b> |
| Function description                        | Receiver Enable (Receiver is enabled and begins searching for a start bit)                 |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>          |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                            |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR1 RE LL_USART_EnableDirectionRx</li> </ul>      |

### **LL\_USART\_DisableDirectionRx**

|                                             |                                                                                             |
|---------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_USART_DisableDirectionRx(USART_TypeDef * USARTx)</code></b> |
| Function description                        | Receiver Disable.                                                                           |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>           |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                             |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR1 RE LL_USART_DisableDirectionRx</li> </ul>      |

**LL\_USART\_EnableDirectionTx**

Function name      **`_STATIC_INLINE void LL_USART_EnableDirectionTx(  
                          USART_TypeDef * USARTx>)`**

Function description      Transmitter Enable.

Parameters      • **USARTx:** USART Instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR1 TE LL\_USART\_EnableDirectionTx

**LL\_USART\_DisableDirectionTx**

Function name      **`_STATIC_INLINE void LL_USART_DisableDirectionTx(  
                          USART_TypeDef * USARTx>)`**

Function description      Transmitter Disable.

Parameters      • **USARTx:** USART Instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR1 TE LL\_USART\_DisableDirectionTx

**LL\_USART\_SetTransferDirection**

Function name      **`_STATIC_INLINE void LL_USART_SetTransferDirection(  
                          USART_TypeDef * USARTx, uint32_t TransferDirection)`**

Function description      Configure simultaneously enabled/disabled states of Transmitter  
and Receiver.

Parameters      • **USARTx:** USART Instance  
• **TransferDirection:** This parameter can be one of the  
following values:  
– LL\_USART\_DIRECTION\_NONE  
– LL\_USART\_DIRECTION\_RX  
– LL\_USART\_DIRECTION\_TX  
– LL\_USART\_DIRECTION\_TX\_RX

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR1 RE LL\_USART\_SetTransferDirection  
• CR1 TE LL\_USART\_SetTransferDirection

**LL\_USART\_GetTransferDirection**

Function name      **`_STATIC_INLINE uint32_t LL_USART_GetTransferDirection(  
                          USART_TypeDef * USARTx)`**

Function description      Return enabled/disabled states of Transmitter and Receiver.

Parameters      • **USARTx:** USART Instance

|                                                   |                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_DIRECTION_NONE</li> <li>– LL_USART_DIRECTION_RX</li> <li>– LL_USART_DIRECTION_TX</li> <li>– LL_USART_DIRECTION_TX_RX</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 RE LL_USART_GetTransferDirection</li> <li>• CR1 TE LL_USART_GetTransferDirection</li> </ul>                                                                                                                                                                    |

### LL\_USART\_SetParity

|                                                   |                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetParity(<br/>  USART_TypeDef * USARTx, uint32_t Parity)</code>                                                                                                                                                                                                                                               |
| Function description                              | Configure Parity (enabled/disabled and parity mode if enabled).                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Parity:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_PARITY_NONE</li> <li>– LL_USART_PARITY_EVEN</li> <li>– LL_USART_PARITY_ODD</li> </ul> </li> </ul>                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• This function selects if hardware parity control (generation and detection) is enabled or disabled. When the parity control is enabled (Odd or Even), computed parity bit is inserted at the MSB position (9th or 8th bit depending on data width) and parity is checked on the received data.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 PS LL_USART_SetParity</li> <li>• CR1 PCE LL_USART_SetParity</li> </ul>                                                                                                                                                                                                                                |

### LL\_USART\_GetParity

|                                                   |                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetParity(<br/>  USART_TypeDef * USARTx)</code>                                                                                                                                                                           |
| Function description                              | Return Parity configuration (enabled/disabled and parity mode if enabled)                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                 |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_PARITY_NONE</li> <li>– LL_USART_PARITY_EVEN</li> <li>– LL_USART_PARITY_ODD</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 PS LL_USART_GetParity</li> <li>• CR1 PCE LL_USART_GetParity</li> </ul>                                                                                                                                               |

**LL\_USART\_SetWakeUpMethod**

|                                                   |                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetWakeUpMethod(<br/>    USART_TypeDef * USARTx, uint32_t Method)</code>                                                                                                                                                                                  |
| Function description                              | Set Receiver Wake Up method from Mute mode.                                                                                                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Method:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_WAKEUP_IDLELINE</li> <li>– LL_USART_WAKEUP_ADDRESSMARK</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 WAKE LL_USART_SetWakeUpMethod</li> </ul>                                                                                                                                                                                                         |

**LL\_USART\_GetWakeUpMethod**

|                                                   |                                                                                                                                                                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetWakeUpMethod(<br/>    USART_TypeDef * USARTx)</code>                                                                                                                                               |
| Function description                              | Return Receiver Wake Up method from Mute mode.                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_WAKEUP_IDLELINE</li> <li>– LL_USART_WAKEUP_ADDRESSMARK</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 WAKE LL_USART_GetWakeUpMethod</li> </ul>                                                                                                                                                         |

**LL\_USART\_SetDataWidth**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetDataWidth(<br/>    USART_TypeDef * USARTx, uint32_t DataWidth)</code>                                                                                                                                                                                                                                                         |
| Function description                              | Set Word length (i.e.                                                                                                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>DataWidth:</b> This parameter can be one of the following values: (*) Values not available on all devices           <ul style="list-style-type: none"> <li>– LL_USART_DATAWIDTH_7B (*)</li> <li>– LL_USART_DATAWIDTH_8B</li> <li>– LL_USART_DATAWIDTH_9B</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 M0 LL_USART_SetDataWidth</li> <li>• CR1 M1 LL_USART_SetDataWidth</li> </ul>                                                                                                                                                                                                                                             |

**LL\_USART\_GetDataWidth**

|                                                   |                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetDataWidth(<br/>    USART_TypeDef * USARTx)</code>                                                                                                                                                                                                                          |
| Function description                              | Return Word length (i.e.                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: (*)<br/>Values not available on all devices           <ul style="list-style-type: none"> <li>- LL_USART_DATAWIDTH_7B (*)</li> <li>- LL_USART_DATAWIDTH_8B</li> <li>- LL_USART_DATAWIDTH_9B</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 M0 LL_USART_GetDataWidth</li> <li>• CR1 M1 LL_USART_GetDataWidth</li> </ul>                                                                                                                                                                                              |

**LL\_USART\_EnableMuteMode**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_EnableMuteMode(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Allow switch between Mute Mode and Active mode.                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 MME LL_USART_EnableMuteMode</li> </ul>        |

**LL\_USART\_DisableMuteMode**

|                                                   |                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_DisableMuteMode(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Prevent Mute Mode use.                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 MME LL_USART_DisableMuteMode</li> </ul>        |

**LL\_USART\_IsEnabledMuteMode**

|                      |                                                                                                   |
|----------------------|---------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledMuteMode(<br/>    USART_TypeDef * USARTx)</code> |
| Function description | Indicate if switch between Mute Mode and Active mode is allowed.                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                 |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                |

- Reference Manual to LL API cross reference:
- CR1 MME LL\_USART\_IsEnabledMuteMode

### LL\_USART\_SetOverSampling

|                                             |                                                                                                                                                                                                                                                                                       |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_USART_SetOverSampling(USART_TypeDef * USARTx, uint32_t OverSampling)</code>                                                                                                                                                                             |
| Function description                        | Set Oversampling to 8-bit or 16-bit mode.                                                                                                                                                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>OverSampling:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_USART_OVERSAMPLING_16</li> <li>– LL_USART_OVERSAMPLING_8</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                       |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR1 OVER8 LL_USART_SetOverSampling</li> </ul>                                                                                                                                                                                                |

### LL\_USART\_GetOverSampling

|                                             |                                                                                                                                                                                                                                 |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_USART_GetOverSampling(USART_TypeDef * USARTx)</code>                                                                                                                                          |
| Function description                        | Return Oversampling mode.                                                                                                                                                                                                       |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                               |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_USART_OVERSAMPLING_16</li> <li>– LL_USART_OVERSAMPLING_8</li> </ul> </li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR1 OVER8 LL_USART_GetOverSampling</li> </ul>                                                                                                                                          |

### LL\_USART\_SetLastClkPulseOutput

|                      |                                                                                                                                                                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_USART_SetLastClkPulseOutput(USART_TypeDef * USARTx, uint32_t LastBitClockPulse)</code>                                                                                                                                                                                   |
| Function description | Configure if Clock pulse of the last data bit is output to the SCLK pin or not.                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>LastBitClockPulse:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_USART_LASTCLKPULSE_NO_OUTPUT</li> <li>– LL_USART_LASTCLKPULSE_OUTPUT</li> </ul> </li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_USART_INSTANCE(USARTx) can be used to check whether or not Synchronous mode is supported by the USARTx instance.</li> </ul>                                                                                                                          |

- Reference Manual to  
LL API cross  
reference:
- CR2 LBCL LL\_USART\_SetLastClkPulseOutput

### **LL\_USART\_GetLastClkPulseOutput**

|                                                   |                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t<br/>LL_USART_GetLastClkPulseOutput (USART_TypeDef *<br/>USARTx)</code>                                                                                                                                                 |
| Function description                              | Retrieve Clock pulse of the last data bit output configuration (Last bit Clock pulse output to the SCLK pin or not)                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_LASTCLKPULSE_NO_OUTPUT</li> <li>– LL_USART_LASTCLKPULSE_OUTPUT</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_USART_INSTANCE(USARTx) can be used to check whether or not Synchronous mode is supported by the USARTx instance.</li> </ul>                                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 LBCL LL_USART_GetLastClkPulseOutput</li> </ul>                                                                                                                                                           |

### **LL\_USART\_SetClockPhase**

|                                                   |                                                                                                                                                                                                                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetClockPhase<br/>(USART_TypeDef * USARTx, uint32_t ClockPhase)</code>                                                                                                                                                                             |
| Function description                              | Select the phase of the clock output on the SCLK pin in synchronous mode.                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>ClockPhase:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_PHASE_1EDGE</li> <li>– LL_USART_PHASE_2EDGE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                        |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_USART_INSTANCE(USARTx) can be used to check whether or not Synchronous mode is supported by the USARTx instance.</li> </ul>                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CPHA LL_USART_SetClockPhase</li> </ul>                                                                                                                                                                                                    |

### **LL\_USART\_GetClockPhase**

|                      |                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_USART_GetClockPhase<br/>(USART_TypeDef * USARTx)</code> |
| Function description | Return phase of the clock output on the SCLK pin in synchronous mode.                     |

|                                                   |                                                                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_PHASE_1EDGE</li> <li>– LL_USART_PHASE_2EDGE</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_USART_INSTANCE(USARTx) can be used to check whether or not Synchronous mode is supported by the USARTx instance.</li> </ul>                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CPHA LL_USART_GetClockPhase</li> </ul>                                                                                                                                                |

### LL\_USART\_SetClockPolarity

|                                                   |                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><u>STATIC_INLINE void LL_USART_SetClockPolarity(USART_TypeDef * USARTx, uint32_t ClockPolarity)</u></b>                                                                                                                                                                                   |
| Function description                              | Select the polarity of the clock output on the SCLK pin in synchronous mode.                                                                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>ClockPolarity:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_POLARITY_LOW</li> <li>– LL_USART_POLARITY_HIGH</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_USART_INSTANCE(USARTx) can be used to check whether or not Synchronous mode is supported by the USARTx instance.</li> </ul>                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CPOL LL_USART_SetClockPolarity</li> </ul>                                                                                                                                                                                                       |

### LL\_USART\_GetClockPolarity

|                                                   |                                                                                                                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><u>STATIC_INLINE uint32_t LL_USART_GetClockPolarity(USART_TypeDef * USARTx)</u></b>                                                                                                                                                |
| Function description                              | Return polarity of the clock output on the SCLK pin in synchronous mode.                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_POLARITY_LOW</li> <li>– LL_USART_POLARITY_HIGH</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_USART_INSTANCE(USARTx) can be used to check whether or not Synchronous mode is supported by the USARTx instance.</li> </ul>                                                         |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CPOL LL_USART_GetClockPolarity</li> </ul>                                                                                                                                                |

**LL\_USART\_ConfigClock**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_USART_ConfigClock<br/>(USART_TypeDef * USARTx, uint32_t Phase, uint32_t Polarity,<br/>uint32_t LBPCOutput)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Function description                              | Configure Clock signal format (Phase Polarity and choice about output of last bit clock pulse)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Phase:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_USART_PHASE_1EDGE</li> <li>– LL_USART_PHASE_2EDGE</li> </ul> </li> <li>• <b>Polarity:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_USART_POLARITY_LOW</li> <li>– LL_USART_POLARITY_HIGH</li> </ul> </li> <li>• <b>LBPCOutput:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_USART_LASTCLKPULSE_NO_OUTPUT</li> <li>– LL_USART_LASTCLKPULSE_OUTPUT</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_USART_INSTANCE(USARTx) can be used to check whether or not Synchronous mode is supported by the USARTx instance.</li> <li>• Call of this function is equivalent to following function call sequence : Clock Phase configuration using LL_USART_SetClockPhase() functionClock Polarity configuration using LL_USART_SetClockPolarity() functionOutput of Last bit Clock pulse configuration using LL_USART_SetLastClkPulseOutput() function</li> </ul>                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CPHA LL_USART_ConfigClock</li> <li>• CR2 CPOL LL_USART_ConfigClock</li> <li>• CR2 LBCL LL_USART_ConfigClock</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

**LL\_USART\_EnableSCLKOutput**

|                                                   |                                                                                                                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>_STATIC_INLINE void LL_USART_EnableSCLKOutput<br/>(USART_TypeDef * USARTx)</code>                                                                                       |
| Function description                              | Enable Clock output on SCLK pin.                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_USART_INSTANCE(USARTx) can be used to check whether or not Synchronous mode is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CLKEN LL_USART_EnableSCLKOutput</li> </ul>                                                                                       |

**LL\_USART\_DisableSCLKOutput**

|                                                   |                                                                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_DisableSCLKOutput(<br/>    USART_TypeDef * USARTx)</code>                                                                                              |
| Function description                              | Disable Clock output on SCLK pin.                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro <code>IS_USART_INSTANCE(USARTx)</code> can be used to check whether or not Synchronous mode is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CLKEN LL_USART_DisableSCLKOutput</li> </ul>                                                                                                   |

**LL\_USART\_IsEnabledSCLKOutput**

|                                                   |                                                                                                                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledSCLKOutput(<br/>    USART_TypeDef * USARTx)</code>                                                                                        |
| Function description                              | Indicate if Clock output on SCLK pin is enabled.                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro <code>IS_USART_INSTANCE(USARTx)</code> can be used to check whether or not Synchronous mode is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 CLKEN LL_USART_IsEnabledSCLKOutput</li> </ul>                                                                                                 |

**LL\_USART\_SetStopBitsLength**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetStopBitsLength(<br/>    USART_TypeDef * USARTx, uint32_t StopBits)</code>                                                                                                                                                                                                                                                                                         |
| Function description                              | Set the length of the stop bits.                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>StopBits:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– <code>LL_USART_STOPBITS_0_5</code></li> <li>– <code>LL_USART_STOPBITS_1</code></li> <li>– <code>LL_USART_STOPBITS_1_5</code></li> <li>– <code>LL_USART_STOPBITS_2</code></li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 STOP LL_USART_SetStopBitsLength</li> </ul>                                                                                                                                                                                                                                                                                                                  |

**LL\_USART\_GetStopBitsLength**

|                                                   |                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetStopBitsLength(<br/>    USART_TypeDef * USARTx)</code>                                                                                                                                                                                                    |
| Function description                              | Retrieve the length of the stop bits.                                                                                                                                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_USART_STOPBITS_0_5</li> <li>- LL_USART_STOPBITS_1</li> <li>- LL_USART_STOPBITS_1_5</li> <li>- LL_USART_STOPBITS_2</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 STOP LL_USART_GetStopBitsLength</li> </ul>                                                                                                                                                                                                              |

**LL\_USART\_ConfigCharacter**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_ConfigCharacter(<br/>    USART_TypeDef * USARTx, uint32_t DataWidth, uint32_t<br/>    Parity, uint32_t StopBits)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Function description                              | Configure Character frame format (Datawidth, Parity control, Stop Bits)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>DataWidth:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_USART_DATAWIDTH_7B (*)</li> <li>- LL_USART_DATAWIDTH_8B</li> <li>- LL_USART_DATAWIDTH_9B</li> </ul> </li> <li>• <b>Parity:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>- LL_USART_PARITY_NONE</li> <li>- LL_USART_PARITY_EVEN</li> <li>- LL_USART_PARITY_ODD</li> </ul> </li> <li>• <b>StopBits:</b> This parameter can be one of the following values:<br/>           (*) Values not available on all devices             <ul style="list-style-type: none"> <li>- LL_USART_STOPBITS_0_5</li> <li>- LL_USART_STOPBITS_1</li> <li>- LL_USART_STOPBITS_1_5</li> <li>- LL_USART_STOPBITS_2</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• Call of this function is equivalent to following function call sequence : Data Width configuration using LL_USART_SetDataWidth() functionParity Control and mode configuration using LL_USART_SetParity() functionStop bits configuration using LL_USART_SetStopBitsLength() function</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 PS LL_USART_ConfigCharacter</li> <li>• CR1 PCE LL_USART_ConfigCharacter</li> <li>• CR1 M0 LL_USART_ConfigCharacter</li> <li>• CR1 M1 LL_USART_ConfigCharacter</li> <li>• CR2 STOP LL_USART_ConfigCharacter</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

**LL\_USART\_SetTXRXSwap**

|                                                   |                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetTXRXSwap(<br/>    USART_TypeDef * USARTx, uint32_t SwapConfig)</code>                                                                                                                                                                                |
| Function description                              | Configure TX/RX pins swapping setting.                                                                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>SwapConfig:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_USART_TXRX_STANDARD</li> <li>– LL_USART_TXRX_SWAPPED</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 SWAP LL_USART_SetTXRXSwap</li> </ul>                                                                                                                                                                                                           |

**LL\_USART\_GetTXRXSwap**

|                                                   |                                                                                                                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetTXRXSwap(<br/>    USART_TypeDef * USARTx)</code>                                                                                                                                             |
| Function description                              | Retrieve TX/RX pins swapping configuration.                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_USART_TXRX_STANDARD</li> <li>– LL_USART_TXRX_SWAPPED</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 SWAP LL_USART_GetTXRXSwap</li> </ul>                                                                                                                                                       |

**LL\_USART\_SetRXPinLevel**

|                                                   |                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetRXPinLevel(<br/>    USART_TypeDef * USARTx, uint32_t PinInvMethod)</code>                                                                                                                                                                                             |
| Function description                              | Configure RX pin active level logic.                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>PinInvMethod:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_USART_RXPIN_LEVEL_STANDARD</li> <li>– LL_USART_RXPIN_LEVEL_INVERTED</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 RXINV LL_USART_SetRXPinLevel</li> </ul>                                                                                                                                                                                                                         |

**LL\_USART\_GetRXPinLevel**

Function name **`_STATIC_INLINE uint32_t LL_USART_GetRXPinLevel(  
USART_TypeDef * USARTx)`**

Function description Retrieve RX pin active level logic configuration.

Parameters • **USARTx:** USART Instance

Return values • **Returned:** value can be one of the following values:  
– LL\_USART\_RXPIN\_LEVEL\_STANDARD  
– LL\_USART\_RXPIN\_LEVEL\_INVERTED

Reference Manual to  
LL API cross  
reference:  
• CR2 RXINV LL\_USART\_GetRXPinLevel

**LL\_USART\_SetTXPinLevel**

Function name **`_STATIC_INLINE void LL_USART_SetTXPinLevel(  
USART_TypeDef * USARTx, uint32_t PinInvMethod)`**

Function description Configure TX pin active level logic.

Parameters • **USARTx:** USART Instance  
• **PinInvMethod:** This parameter can be one of the following values:  
– LL\_USART\_TXPIN\_LEVEL\_STANDARD  
– LL\_USART\_TXPIN\_LEVEL\_INVERTED

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
• CR2 TXINV LL\_USART\_SetTXPinLevel

**LL\_USART\_GetTXPinLevel**

Function name **`_STATIC_INLINE uint32_t LL_USART_GetTXPinLevel(  
USART_TypeDef * USARTx)`**

Function description Retrieve TX pin active level logic configuration.

Parameters • **USARTx:** USART Instance

Return values • **Returned:** value can be one of the following values:  
– LL\_USART\_TXPIN\_LEVEL\_STANDARD  
– LL\_USART\_TXPIN\_LEVEL\_INVERTED

Reference Manual to  
LL API cross  
reference:  
• CR2 TXINV LL\_USART\_GetTXPinLevel

**LL\_USART\_SetBinaryDataLogic**

Function name **`_STATIC_INLINE void LL_USART_SetBinaryDataLogic(  
USART_TypeDef * USARTx, uint32_t DataLogic)`**

Function description Configure Binary data logic.

Parameters • **USARTx:** USART Instance

- **DataLogic:** This parameter can be one of the following values:
  - LL\_USART\_BINARY\_LOGIC\_POSITIVE
  - LL\_USART\_BINARY\_LOGIC\_NEGATIVE
- **None**
- Notes
  - Allow to define how Logical data from the data register are send/received : either in positive/direct logic (1=H, 0=L) or in negative/inverse logic (1=L, 0=H)
- Reference Manual to LL API cross reference:
  - CR2 DATAINV LL\_USART\_SetBinaryDataLogic

### **LL\_USART\_GetBinaryDataLogic**

- Function name      **`__STATIC_INLINE uint32_t LL_USART_GetBinaryDataLogic(  
                  USART_TypeDef * USARTx)`**
- Function description      Retrieve Binary data configuration.
- Parameters
  - **USARTx:** USART Instance
- Return values
  - **Returned:** value can be one of the following values:
    - LL\_USART\_BINARY\_LOGIC\_POSITIVE
    - LL\_USART\_BINARY\_LOGIC\_NEGATIVE
- Reference Manual to LL API cross reference:
  - CR2 DATAINV LL\_USART\_GetBinaryDataLogic

### **LL\_USART\_SetTransferBitOrder**

- Function name      **`__STATIC_INLINE void LL_USART_SetTransferBitOrder(  
                  USART_TypeDef * USARTx, uint32_t BitOrder)`**
- Function description      Configure transfer bit order (either Less or Most Significant Bit First)
- Parameters
  - **USARTx:** USART Instance
  - **BitOrder:** This parameter can be one of the following values:
    - LL\_USART\_BITORDER\_LSBFIRST
    - LL\_USART\_BITORDER\_MSBFIRST
- Return values
  - **None**
- Notes
  - MSB First means data is transmitted/received with the MSB first, following the start bit. LSB First means data is transmitted/received with data bit 0 first, following the start bit.
- Reference Manual to LL API cross reference:
  - CR2 MSBFIRST LL\_USART\_SetTransferBitOrder

**LL\_USART\_GetTransferBitOrder**

|                                                   |                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetTransferBitOrder(<br/>    USART_TypeDef * USARTx)</code>                                                                                                                                              |
| Function description                              | Return transfer bit order (either Less or Most Significant Bit First)                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_USART_BITORDER_LSBFIRST</li> <li>– LL_USART_BITORDER_MSBFIRST</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• MSB First means data is transmitted/received with the MSB first, following the start bit. LSB First means data is transmitted/received with data bit 0 first, following the start bit.</li> </ul>       |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 MSBFIRST LL_USART_GetTransferBitOrder</li> </ul>                                                                                                                                                    |

**LL\_USART\_EnableAutoBaudRate**

|                                                   |                                                                                                                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_EnableAutoBaudRate(<br/>    USART_TypeDef * USARTx)</code>                                                                                                                                        |
| Function description                              | Enable Auto Baud-Rate Detection.                                                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro<br/><code>IS_USART_AUTOBAUDRATE_DETECTION_INSTANCE(USARTx)</code> can be used to check whether or not Auto Baud Rate detection feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual<br>to LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 ABREN LL_USART_EnableAutoBaudRate</li> </ul>                                                                                                                                             |

**LL\_USART\_DisableAutoBaudRate**

|                                                   |                                                                                                                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_DisableAutoBaudRate(<br/>    USART_TypeDef * USARTx)</code>                                                                                                                                       |
| Function description                              | Disable Auto Baud-Rate Detection.                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro<br/><code>IS_USART_AUTOBAUDRATE_DETECTION_INSTANCE(USARTx)</code> can be used to check whether or not Auto Baud Rate detection feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual<br>to LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 ABREN LL_USART_DisableAutoBaudRate</li> </ul>                                                                                                                                            |

**LL\_USART\_IsEnabledAutoBaud**

|                                             |                                                                                                                                                                                                                                       |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE uint32_t LL_USART_IsEnabledAutoBaud<br/>(USART_TypeDef * USARTx)</code>                                                                                                                                          |
| Function description                        | Indicate if Auto Baud-Rate Detection mechanism is enabled.                                                                                                                                                                            |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                     |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                    |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro<br/><code>IS_USART_AUTOBAUDRATE_DETECTION_INSTANCE(USARTx)</code> can be used to check whether or not Auto Baud Rate detection feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR2 ABREN LL_USART_IsEnabledAutoBaud</li> </ul>                                                                                                                                              |

**LL\_USART\_SetAutoBaudRateMode**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE void LL_USART_SetAutoBaudRateMode<br/>(USART_TypeDef * USARTx, uint32_t AutoBaudRateMode)</code>                                                                                                                                                                                                                                                                                                                                                     |
| Function description                        | Set Auto Baud-Rate mode bits.                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>AutoBaudRateMode:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_USART_AUTOBAUD_DETECT_ON_STARTBIT</code></li> <li>– <code>LL_USART_AUTOBAUD_DETECT_ON_FALLINGEDGE</code></li> <li>– <code>LL_USART_AUTOBAUD_DETECT_ON_7F_FRAME</code></li> <li>– <code>LL_USART_AUTOBAUD_DETECT_ON_55_FRAME</code></li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                           |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro<br/><code>IS_USART_AUTOBAUDRATE_DETECTION_INSTANCE(USARTx)</code> can be used to check whether or not Auto Baud Rate detection feature is supported by the USARTx instance.</li> </ul>                                                                                                                                                                                                                                     |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR2 ABRMODE LL_USART_SetAutoBaudRateMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                              |

**LL\_USART\_GetAutoBaudRateMode**

|                      |                                                                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE uint32_t LL_USART_GetAutoBaudRateMode<br/>(USART_TypeDef * USARTx)</code>                                                                                                                                                                                        |
| Function description | Return Auto Baud-Rate mode.                                                                                                                                                                                                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– <code>LL_USART_AUTOBAUD_DETECT_ON_STARTBIT</code></li> <li>– <code>LL_USART_AUTOBAUD_DETECT_ON_FALLINGEDGE</code></li> </ul> </li> </ul> |

|                                             |                                                                                                                                                                                                                           |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                             | <ul style="list-style-type: none"> <li>- LL_USART_AUTOBAUD_DETECT_ON_7F_FRAME</li> <li>- LL_USART_AUTOBAUD_DETECT_ON_55_FRAME</li> </ul>                                                                                  |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro<br/>IS_USART_AUTOBAUDRATE_DETECTION_INSTANCE(USART x) can be used to check whether or not Auto Baud Rate detection feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR2 ABRMODE LL_USART_GetAutoBaudRateMode</li> </ul>                                                                                                                              |

### LL\_USART\_EnableRxTimeout

|                      |                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_USART_EnableRxTimeout(<br/>    USART_TypeDef * USARTx)</code> |
| Function description | Enable Receiver Timeout.                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                             |

Reference Manual to LL API cross reference:

- CR2 RTOEN LL\_USART\_EnableRxTimeout

### LL\_USART\_DisableRxTimeout

|                      |                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_USART_DisableRxTimeout(<br/>    USART_TypeDef * USARTx)</code> |
| Function description | Disable Receiver Timeout.                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>            |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                              |

Reference Manual to LL API cross reference:

- CR2 RTOEN LL\_USART\_DisableRxTimeout

### LL\_USART\_IsEnabledRxTimeout

|                      |                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledRxTimeout(<br/>    USART_TypeDef * USARTx)</code> |
| Function description | Indicate if Receiver Timeout feature is enabled.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                 |

Reference Manual to LL API cross reference:

- CR2 RTOEN LL\_USART\_IsEnabledRxTimeout

**LL\_USART\_ConfigNodeAddress**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_ConfigNodeAddress<br/>(USART_TypeDef * USARTx, uint32_t AddressLen, uint32_t<br/>NodeAddress)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Set Address of the USART node.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>AddressLen:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_USART_ADDRESS_DETECT_4B</li> <li>– LL_USART_ADDRESS_DETECT_7B</li> </ul> </li> <li>• <b>NodeAddress:</b> 4 or 7 bit Address of the USART node.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• This is used in multiprocessor communication during Mute mode or Stop mode, for wake up with address mark detection.</li> <li>• 4bits address node is used when 4-bit Address Detection is selected in ADDM7. (b7-b4 should be set to 0) 8bits address node is used when 7-bit Address Detection is selected in ADDM7. (This is used in multiprocessor communication during Mute mode or Stop mode, for wake up with 7-bit address mark detection. The MSB of the character sent by the transmitter should be equal to 1. It may also be used for character detection during normal reception, Mute mode inactive (for example, end of block detection in ModBus protocol). In this case, the whole received character (8-bit) is compared to the ADD[7:0] value and CMF flag is set on match)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 ADD LL_USART_ConfigNodeAddress</li> <li>• CR2 ADDM7 LL_USART_ConfigNodeAddress</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

**LL\_USART\_GetNodeAddress**

|                                                   |                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetNodeAddress<br/>(USART_TypeDef * USARTx)</code>                                                                                                                                                                                                                    |
| Function description                              | Return 8 bit Address of the USART node as set in ADD field of CR2.                                                                                                                                                                                                                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Address:</b> of the USART node (Value between Min_Data=0 and Max_Data=255)</li> </ul>                                                                                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• If 4-bit Address Detection is selected in ADDM7, only 4bits (b3-b0) of returned value are relevant (b31-b4 are not relevant) If 7-bit Address Detection is selected in ADDM7, only 8bits (b7-b0) of returned value are relevant (b31-b8 are not relevant)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 ADD LL_USART_GetNodeAddress</li> </ul>                                                                                                                                                                                                                           |

**LL\_USART\_GetNodeAddressLen**

|                                                   |                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetNodeAddressLen(<br/>    USART_TypeDef * USARTx)</code>                                                                                                                                                |
| Function description                              | Return Length of Node Address used in Address Detection mode<br>(7-bit or 4-bit)                                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:             <ul style="list-style-type: none"> <li>– LL_USART_ADDRESS_DETECT_4B</li> <li>– LL_USART_ADDRESS_DETECT_7B</li> </ul> </li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 ADDM7 LL_USART_GetNodeAddressLen</li> </ul>                                                                                                                                                         |

**LL\_USART\_EnableRTSHWFlowCtrl**

|                                                   |                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_EnableRTSHWFlowCtrl(<br/>    USART_TypeDef * USARTx)</code>                                                                                                  |
| Function description                              | Enable RTS HW Flow Control.                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_HWFLOW_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 RTSE LL_USART_EnableRTSHWFlowCtrl</li> </ul>                                                                                                        |

**LL\_USART\_DisableRTSHWFlowCtrl**

|                                                   |                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_DisableRTSHWFlowCtrl(<br/>    USART_TypeDef * USARTx)</code>                                                                                                 |
| Function description                              | Disable RTS HW Flow Control.                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_HWFLOW_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 RTSE LL_USART_DisableRTSHWFlowCtrl</li> </ul>                                                                                                       |

**LL\_USART\_EnableCTSHWFlowCtrl**

|                                                   |                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_EnableCTSHWFlowCtrl(<br/>USART_TypeDef * USARTx)</code></b>                                                                                                |
| Function description                              | Enable CTS HW Flow Control.                                                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_HWFLOW_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 CTSE LL_USART_EnableCTSHWFlowCtrl</li> </ul>                                                                                                        |

**LL\_USART\_DisableCTSHWFlowCtrl**

|                                                   |                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_DisableCTSHWFlowCtrl(<br/>USART_TypeDef * USARTx)</code></b>                                                                                               |
| Function description                              | Disable CTS HW Flow Control.                                                                                                                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_HWFLOW_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 CTSE LL_USART_DisableCTSHWFlowCtrl</li> </ul>                                                                                                       |

**LL\_USART\_SetHWFlowCtrl**

|                                     |                                                                                                                                                                                                                                                                                                                                                                    |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                       | <b><code>_STATIC_INLINE void LL_USART_SetHWFlowCtrl(<br/>USART_TypeDef * USARTx, uint32_t HardwareFlowControl)</code></b>                                                                                                                                                                                                                                          |
| Function description                | Configure HW Flow Control mode (both CTS and RTS)                                                                                                                                                                                                                                                                                                                  |
| Parameters                          | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>HardwareFlowControl:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_USART_HWCONTROL_NONE</li> <li>– LL_USART_HWCONTROL_RTS</li> <li>– LL_USART_HWCONTROL_CTS</li> <li>– LL_USART_HWCONTROL_RTS_CTS</li> </ul> </li> </ul> |
| Return values                       | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                    |
| Notes                               | <ul style="list-style-type: none"> <li>• Macro IS_UART_HWFLOW_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.</li> </ul>                                                                                                                                                                   |
| Reference Manual to<br>LL API cross | <ul style="list-style-type: none"> <li>• CR3 RTSE LL_USART_SetHWFlowCtrl</li> <li>• CR3 CTSE LL_USART_SetHWFlowCtrl</li> </ul>                                                                                                                                                                                                                                     |

reference:

### **LL\_USART\_GetHWFlowCtrl**

|                                                   |                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_USART_GetHWFlowCtrl(<br/>    USART_TypeDef * USARTx)</code></b>                                                                                                                                                                                                            |
| Function description                              | Return HW Flow Control configuration (both CTS and RTS)                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>- LL_USART_HWCONTROL_NONE</li> <li>- LL_USART_HWCONTROL_RTS</li> <li>- LL_USART_HWCONTROL_CTS</li> <li>- LL_USART_HWCONTROL_RTS_CTS</li> </ul> </li> </ul> |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_HWFLOW_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.</li> </ul>                                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 RTSE LL_USART_GetHWFlowCtrl</li> <li>• CR3 CTSE LL_USART_GetHWFlowCtrl</li> </ul>                                                                                                                                                                                  |

### **LL\_USART\_EnableOneBitSamp**

|                                                   |                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_USART_EnableOneBitSamp(<br/>    USART_TypeDef * USARTx)</code></b> |
| Function description                              | Enable One bit sampling method.                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                   |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 ONEBIT LL_USART_EnableOneBitSamp</li> </ul>            |

### **LL\_USART\_DisableOneBitSamp**

|                                                   |                                                                                                      |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_USART_DisableOneBitSamp(<br/>    USART_TypeDef * USARTx)</code></b> |
| Function description                              | Disable One bit sampling method.                                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 ONEBIT LL_USART_DisableOneBitSamp</li> </ul>            |

**LL\_USART\_IsEnabledOneBitSamp**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledOneBitSamp(USART_TypeDef * USARTx)</code> |
| Function description                              | Indicate if One bit sampling method is enabled.                                            |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>USARTx:</b> USART Instance</li></ul>            |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR3 ONEBIT LL_USART_IsEnabledOneBitSamp</li></ul>  |

**LL\_USART\_EnableOverrunDetect**

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_EnableOverrunDetect(USART_TypeDef * USARTx)</code>    |
| Function description                              | Enable Overrun detection.                                                                 |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>USARTx:</b> USART Instance</li></ul>           |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR3 OVRDIS LL_USART_EnableOverrunDetect</li></ul> |

**LL\_USART\_DisableOverrunDetect**

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_DisableOverrunDetect(USART_TypeDef * USARTx)</code>    |
| Function description                              | Disable Overrun detection.                                                                 |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>USARTx:</b> USART Instance</li></ul>            |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR3 OVRDIS LL_USART_DisableOverrunDetect</li></ul> |

**LL\_USART\_IsEnabledOverrunDetect**

|                                                   |                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledOverrunDetect(USART_TypeDef * USARTx)</code> |
| Function description                              | Indicate if Overrun detection is enabled.                                                     |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>USARTx:</b> USART Instance</li></ul>               |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>State:</b> of bit (1 or 0).</li></ul>              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• CR3 OVRDIS LL_USART_IsEnabledOverrunDetect</li></ul>  |

**LL\_USART\_SetWKUPTYPE**

|                                             |                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE void LL_USART_SetWKUPTYPE<br/>(USART_TypeDef * USARTx, uint32_t Type)</code>                                                                                                                                                                                                                      |
| Function description                        | Select event type for Wake UP Interrupt Flag (WUS[1:0] bits)                                                                                                                                                                                                                                                           |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Type:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>– LL_USART_WAKEUP_ON_ADDRESS</li> <li>– LL_USART_WAKEUP_ON_STARTBIT</li> <li>– LL_USART_WAKEUP_ON_RXNE</li> </ul> </li> </ul> |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                        |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro IS_UART_WAKEUP_FROMSTOP_INSTANCE(USARTx) can be used to check whether or not Wake-up from Stop mode feature is supported by the USARTx instance.</li> </ul>                                                                                                             |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR3 WUS LL_USART_SetWKUPTYPE</li> </ul>                                                                                                                                                                                                                                       |

**LL\_USART\_GetWKUPTYPE**

|                                             |                                                                                                                                                                                                                                                                          |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>_STATIC_INLINE uint32_t LL_USART_GetWKUPTYPE<br/>(USART_TypeDef * USARTx)</code>                                                                                                                                                                                   |
| Function description                        | Return event type for Wake UP Interrupt Flag (WUS[1:0] bits)                                                                                                                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                        |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values: <ul style="list-style-type: none"> <li>– LL_USART_WAKEUP_ON_ADDRESS</li> <li>– LL_USART_WAKEUP_ON_STARTBIT</li> <li>– LL_USART_WAKEUP_ON_RXNE</li> </ul> </li> </ul> |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro IS_UART_WAKEUP_FROMSTOP_INSTANCE(USARTx) can be used to check whether or not Wake-up from Stop mode feature is supported by the USARTx instance.</li> </ul>                                                               |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR3 WUS LL_USART_GetWKUPTYPE</li> </ul>                                                                                                                                                                                         |

**LL\_USART\_SetBaudRate**

|                      |                                                                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>_STATIC_INLINE void LL_USART_SetBaudRate<br/>(USART_TypeDef * USARTx, uint32_t PeriphClk, uint32_t OverSampling, uint32_t BaudRate)</code>                                                                   |
| Function description | Configure USART BRR register for achieving expected Baud Rate value.                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>PeriphClk:</b> Peripheral Clock</li> <li>• <b>OverSampling:</b> This parameter can be one of the following values:</li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | <ul style="list-style-type: none"> <li>- LL_USART_OVERSAMPLING_16</li> <li>- LL_USART_OVERSAMPLING_8</li> </ul>                                                                                                                                                                                                                                                                                                                      |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>BaudRate:</b> Baud Rate</li> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                |
| Notes                                             | <ul style="list-style-type: none"> <li>• Compute and set USARTDIV value in BRR Register (full BRR content) according to used Peripheral Clock, Oversampling mode, and expected Baud Rate values</li> <li>• Peripheral clock and Baud rate values provided as function parameters should be valid (Baud rate value != 0)</li> <li>• In case of oversampling by 16 and 8, BRR content must be greater than or equal to 16d.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BRR BRR LL_USART_SetBaudRate</li> </ul>                                                                                                                                                                                                                                                                                                                                                     |

### LL\_USART\_GetBaudRate

|                                                   |                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetBaudRate(<br/>USART_TypeDef * USARTx, uint32_t PeriphClk, uint32_t<br/>OverSampling)</code>                                                                                                                                                                                              |
| Function description                              | Return current Baud Rate value, according to USARTDIV present in BRR register (full BRR content), and to used Peripheral Clock and Oversampling mode values.                                                                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>PeriphClk:</b> Peripheral Clock</li> <li>• <b>OverSampling:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_USART_OVERSAMPLING_16</li> <li>- LL_USART_OVERSAMPLING_8</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Baud:</b> Rate</li> </ul>                                                                                                                                                                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• In case of non-initialized or invalid value stored in BRR register, value 0 will be returned.</li> <li>• In case of oversampling by 16 and 8, BRR content must be greater than or equal to 16d.</li> </ul>                                                                                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• BRR BRR LL_USART_GetBaudRate</li> </ul>                                                                                                                                                                                                                                                    |

### LL\_USART\_SetRxTimeout

|                      |                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_USART_SetRxTimeout(<br/>USART_TypeDef * USARTx, uint32_t Timeout)</code>                                                           |
| Function description | Set Receiver Time Out Value (expressed in nb of bits duration)                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Timeout:</b> Value between Min_Data=0x00 and Max_Data=0x00FFFFFF</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                  |

- Reference Manual to  
LL API cross  
reference:
- RTOR RTO LL\_USART\_SetRxTimeout

### **LL\_USART\_GetRxTimeout**

|                                                   |                                                                                                                 |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetRxTimeout(<br/>    USART_TypeDef * USARTx)</code>                    |
| Function description                              | Get Receiver Time Out Value (expressed in nb of bits duration)                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x00FFFFFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• RTOR RTO LL_USART_SetRxTimeout</li> </ul>                              |

### **LL\_USART\_SetBlockLength**

|                                                   |                                                                                                                                                                |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetBlockLength(<br/>    USART_TypeDef * USARTx, uint32_t BlockLength)</code>                                               |
| Function description                              | Set Block Length value in reception.                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>BlockLength:</b> Value between Min_Data=0x00 and Max_Data=0xFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• RTOR BLEN LL_USART_SetBlockLength</li> </ul>                                                                          |

### **LL\_USART\_GetBlockLength**

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetBlockLength(<br/>    USART_TypeDef * USARTx)</code>            |
| Function description                              | Get Block Length value in reception.                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• RTOR BLEN LL_USART_SetBlockLength</li> </ul>                     |

### **LL\_USART\_EnableIrda**

|                      |                                                                                        |
|----------------------|----------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_USART_EnableIrda(<br/>    USART_TypeDef * USARTx)</code> |
| Function description | Enable IrDA mode.                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>      |

|                                             |                                                                                                                                                                        |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                          |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_IRDA_INSTANCE(USARTx) can be used to check whether or not IrDA feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR3 IREN LL_USART_EnableIrda</li> </ul>                                                                                         |

### LL\_USART\_DisableIrda

|                                             |                                                                                                                                                                        |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_USART_DisableIrda(<br/>    USART_TypeDef * USARTx)</code>                                                                                |
| Function description                        | Disable IrDA mode.                                                                                                                                                     |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                        |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                          |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_IRDA_INSTANCE(USARTx) can be used to check whether or not IrDA feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR3 IREN LL_USART_DisableIrda</li> </ul>                                                                                        |

### LL\_USART\_IsEnabledIrda

|                                             |                                                                                                                                                                        |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledIrda(<br/>    USART_TypeDef * USARTx)</code>                                                                          |
| Function description                        | Indicate if IrDA mode is enabled.                                                                                                                                      |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                        |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                       |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_IRDA_INSTANCE(USARTx) can be used to check whether or not IrDA feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR3 IREN LL_USART_IsEnabledIrda</li> </ul>                                                                                      |

### LL\_USART\_SetIrdaPowerMode

|                      |                                                                                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_USART_SetIrdaPowerMode(<br/>    USART_TypeDef * USARTx, uint32_t PowerMode)</code>                                                                                                                                                                                                       |
| Function description | Configure IrDA Power Mode (Normal or Low Power)                                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> <li><b>PowerMode:</b> This parameter can be one of the following values:             <ul style="list-style-type: none"> <li>– <code>LL_USART_IRDA_POWER_NORMAL</code></li> <li>– <code>LL_USART_IRDA_POWER_LOW</code></li> </ul> </li> </ul> |

|                                             |                                                                                                                                                                        |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                          |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_IRDA_INSTANCE(USARTx) can be used to check whether or not IrDA feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR3 IRLP LL_USART_SetIrdaPowerMode</li> </ul>                                                                                   |

### LL\_USART\_GetIrdaPowerMode

|                                             |                                                                                                                                                                                                                                        |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><u>STATIC_INLINE uint32_t LL_USART_GetIrdaPowerMode(USART_TypeDef * USARTx)</u></b>                                                                                                                                                 |
| Function description                        | Retrieve IrDA Power Mode configuration (Normal or Low Power)                                                                                                                                                                           |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                        |
| Return values                               | <ul style="list-style-type: none"> <li><b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_IRDA_POWER_NORMAL</li> <li>– LL_USART_PHASE_2EDGE</li> </ul> </li> </ul> |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_IRDA_INSTANCE(USARTx) can be used to check whether or not IrDA feature is supported by the USARTx instance.</li> </ul>                                                                 |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR3 IRLP LL_USART_GetIrdaPowerMode</li> </ul>                                                                                                                                                   |

### LL\_USART\_SetIrdaPrescaler

|                                             |                                                                                                                                                                        |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><u>STATIC_INLINE void LL_USART_SetIrdaPrescaler(USART_TypeDef * USARTx, uint32_t PrescalerValue)</u></b>                                                            |
| Function description                        | Set Irda prescaler value, used for dividing the USART clock source to achieve the Irda Low Power frequency (8 bits value)                                              |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> <li><b>PrescalerValue:</b> Value between Min_Data=0x00 and Max_Data=0xFF</li> </ul>          |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                          |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_IRDA_INSTANCE(USARTx) can be used to check whether or not IrDA feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>GTPR PSC LL_USART_SetIrdaPrescaler</li> </ul>                                                                                   |

### LL\_USART\_GetIrdaPrescaler

|                      |                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><u>STATIC_INLINE uint32_t LL_USART_GetIrdaPrescaler(USART_TypeDef * USARTx)</u></b>                                       |
| Function description | Return Irda prescaler value, used for dividing the USART clock source to achieve the Irda Low Power frequency (8 bits value) |

|                                                   |                                                                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>USARTTx:</b> USART Instance</li> </ul>                                                                                         |
| Return values                                     | <ul style="list-style-type: none"> <li><b>Irda:</b> prescaler value (Value between Min_Data=0x00 and Max_Data=0xFF)</li> </ul>                                           |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_IRDA_INSTANCE(USARTTx) can be used to check whether or not IrDA feature is supported by the USARTTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>GTPR PSC LL_USART_GetIrdaPrescaler</li> </ul>                                                                                     |

### LL\_USART\_EnableSmartcardNACK

|                                                   |                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_EnableSmartcardNACK (USART_TypeDef * USARTTx)</code></b>                                                                                     |
| Function description                              | Enable Smartcard NACK transmission.                                                                                                                                                |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>USARTTx:</b> USART Instance</li> </ul>                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_SMARTCARD_INSTANCE(USARTTx) can be used to check whether or not Smartcard feature is supported by the USARTTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR3 NACK LL_USART_EnableSmartcardNACK</li> </ul>                                                                                            |

### LL\_USART\_DisableSmartcardNACK

|                                                   |                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_DisableSmartcardNACK (USART_TypeDef * USARTTx)</code></b>                                                                                    |
| Function description                              | Disable Smartcard NACK transmission.                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>USARTTx:</b> USART Instance</li> </ul>                                                                                                   |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_SMARTCARD_INSTANCE(USARTTx) can be used to check whether or not Smartcard feature is supported by the USARTTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR3 NACK LL_USART_DisableSmartcardNACK</li> </ul>                                                                                           |

### LL\_USART\_IsEnabledSmartcardNACK

|                      |                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_USART_IsEnabledSmartcardNACK (USART_TypeDef * USARTTx)</code></b> |
| Function description | Indicate if Smartcard NACK transmission is enabled.                                                   |
| Parameters           | <ul style="list-style-type: none"> <li><b>USARTTx:</b> USART Instance</li> </ul>                      |

---

|                                                   |                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR3 NACK LL_USART_IsEnabledSmartcardNACK</li> </ul>                                                                                       |

### LL\_USART\_EnableSmartcard

|                                                   |                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_EnableSmartcard(<br/>    USART_TypeDef * USARTx)</code>                                                                                      |
| Function description                              | Enable Smartcard mode.                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR3 SCEN LL_USART_EnableSmartcard</li> </ul>                                                                                              |

### LL\_USART\_DisableSmartcard

|                                                   |                                                                                                                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_DisableSmartcard(<br/>    USART_TypeDef * USARTx)</code>                                                                                     |
| Function description                              | Disable Smartcard mode.                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR3 SCEN LL_USART_DisableSmartcard</li> </ul>                                                                                             |

### LL\_USART\_IsEnabledSmartcard

|                      |                                                                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledSmartcard(<br/>    USART_TypeDef * USARTx)</code>                                                              |
| Function description | Indicate if Smartcard mode is enabled.                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                 |
| Return values        | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                |
| Notes                | <ul style="list-style-type: none"> <li>Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the</li> </ul> |

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | USARTTx instance.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 SCEN LL_USART_IsEnabledSmartcard</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>LL_USART_SetSmartcardAutoRetryCount</b>        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Function name                                     | <b><code>__STATIC_INLINE void<br/>LL_USART_SetSmartcardAutoRetryCount (USART_TypeDef *<br/>USARTx, uint32_t AutoRetryCount)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Function description                              | Set Smartcard Auto-Retry Count value (SCARCNT[2:0] bits)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>AutoRetryCount:</b> Value between Min_Data=0 and Max_Data=7</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> <li>• This bit-field specifies the number of retries in transmit and receive, in Smartcard mode. In transmission mode, it specifies the number of automatic retransmission retries, before generating a transmission error (FE bit set). In reception mode, it specifies the number of erroneous reception trials, before generating a reception error (RXNE and PE bits set)</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 SCARCNT LL_USART_SetSmartcardAutoRetryCount</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

|                                                   |                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                   | Return Smartcard Auto-Retry Count value (SCARCNT[2:0] bits)                                                                                                                        |
| Function name                                     | <b><code>__STATIC_INLINE uint32_t<br/>LL_USART_GetSmartcardAutoRetryCount (USART_TypeDef *<br/>USARTx)</code></b>                                                                  |
| Function description                              | Return Smartcard Auto-Retry Count value (SCARCNT[2:0] bits)                                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Smartcard:</b> Auto-Retry Count value (Value between Min_Data=0 and Max_Data=7)</li> </ul>                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 SCARCNT LL_USART_GetSmartcardAutoRetryCount</li> </ul>                                                                                |

**LL\_USART\_SetSmartcardPrescaler**

|                                                   |                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetSmartcardPrescaler(USART_TypeDef * USARTx, uint32_t PrescalerValue)</code>                                                                  |
| Function description                              | Set Smartcard prescaler value, used for dividing the USART clock source to provide the SMARTCARD Clock (5 bits value)                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>PrescalerValue:</b> Value between Min_Data=0 and Max_Data=31</li> </ul>                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• GTPR PSC LL_USART_SetSmartcardPrescaler</li> </ul>                                                                                        |

**LL\_USART\_GetSmartcardPrescaler**

|                                                   |                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_GetSmartcardPrescaler(USART_TypeDef * USARTx)</code>                                                                                       |
| Function description                              | Return Smartcard prescaler value, used for dividing the USART clock source to provide the SMARTCARD Clock (5 bits value)                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Smartcard:</b> prescaler value (Value between Min_Data=0 and Max_Data=31)</li> </ul>                                                   |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• GTPR PSC LL_USART_GetSmartcardPrescaler</li> </ul>                                                                                        |

**LL\_USART\_SetSmartcardGuardTime**

|                      |                                                                                                                                                                                    |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_USART_SetSmartcardGuardTime(USART_TypeDef * USARTx, uint32_t GuardTime)</code>                                                                       |
| Function description | Set Smartcard Guard time value, expressed in nb of baud clocks periods (GT[7:0] bits : Guard time value)                                                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>GuardTime:</b> Value between Min_Data=0x00 and Max_Data=0xFF</li> </ul>                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                    |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> </ul> |

Reference Manual to  
LL API cross  
reference:

- GTPR GT LL\_USART\_SetSmartcardGuardTime

### **LL\_USART\_GetSmartcardGuardTime**

|                                                   |                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t<br/>LL_USART_GetSmartcardGuardTime (USART_TypeDef *<br/>USARTx)</code></b>                                                                       |
| Function description                              | Return Smartcard Guard time value, expressed in nb of baud clocks periods (GT[7:0] bits : Guard time value)                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Smartcard:</b> Guard time value (Value between Min_Data=0x00 and Max_Data=0xFF)</li> </ul>                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• GTPR GT LL_USART_GetSmartcardGuardTime</li> </ul>                                                                                         |

### **LL\_USART\_EnableHalfDuplex**

|                                                   |                                                                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_USART_EnableHalfDuplex<br/>(USART_TypeDef * USARTx)</code></b>                                                                                         |
| Function description                              | Enable Single Wire Half-Duplex mode.                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_HALFDUPLEX_INSTANCE(USARTx) can be used to check whether or not Half-Duplex mode is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 HDSEL LL_USART_EnableHalfDuplex</li> </ul>                                                                                                 |

### **LL\_USART\_DisableHalfDuplex**

|                      |                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_USART_DisableHalfDuplex<br/>(USART_TypeDef * USARTx)</code></b>                                                                                        |
| Function description | Disable Single Wire Half-Duplex mode.                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                         |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_UART_HALFDUPLEX_INSTANCE(USARTx) can be used to check whether or not Half-Duplex mode is supported by the USARTx instance.</li> </ul> |

- Reference Manual to  
LL API cross  
reference:
- CR3 HDSEL LL\_USART\_DisableHalfDuplex

### **LL\_USART\_IsEnabledHalfDuplex**

|                                                   |                                                                                                                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledHalfDuplex(<br/>    USART_TypeDef * USARTx)</code>                                                                                     |
| Function description                              | Indicate if Single Wire Half-Duplex mode is enabled.                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_HALFDUPLEX_INSTANCE(USARTx) can be used to check whether or not Half-Duplex mode is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 HDSEL LL_USART_IsEnabledHalfDuplex</li> </ul>                                                                                              |

### **LL\_USART\_SetLINBrkDetectionLen**

|                                                   |                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetLINBrkDetectionLen(<br/>    USART_TypeDef * USARTx, uint32_t LINBDLength)</code>                                                                                                                                                                                 |
| Function description                              | Set LIN Break Detection Length.                                                                                                                                                                                                                                                                         |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>LINBDLength:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_LINBREAK_DETECT_10B</li> <li>– LL_USART_LINBREAK_DETECT_11B</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_LIN_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.</li> </ul>                                                                                                                             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 LBDL LL_USART_SetLINBrkDetectionLen</li> </ul>                                                                                                                                                                                                             |

### **LL\_USART\_GetLINBrkDetectionLen**

|                      |                                                                                                                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t<br/>LL_USART_GetLINBrkDetectionLen (USART_TypeDef *<br/>    USARTx)</code>                                                                                                                                          |
| Function description | Return LIN Break Detection Length.                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_LINBREAK_DETECT_10B</li> <li>– LL_USART_LINBREAK_DETECT_11B</li> </ul> </li> </ul> |

|                                             |                                                                                                                                                                           |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_UART_LIN_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR2 LBDL LL_USART_GetLINBrkDetectionLen</li> </ul>                                                                                 |

### LL\_USART\_EnableLIN

|                                             |                                                                                                                                                                           |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_USART_EnableLIN(<br/>    USART_TypeDef * USARTx)</code>                                                                                     |
| Function description                        | Enable LIN mode.                                                                                                                                                          |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                           |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                             |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_UART_LIN_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR2 LINEN LL_USART_EnableLIN</li> </ul>                                                                                            |

### LL\_USART\_DisableLIN

|                                             |                                                                                                                                                                           |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_USART_DisableLIN(<br/>    USART_TypeDef * USARTx)</code>                                                                                    |
| Function description                        | Disable LIN mode.                                                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                           |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                             |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_UART_LIN_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR2 LINEN LL_USART_DisableLIN</li> </ul>                                                                                           |

### LL\_USART\_IsEnabledLIN

|                      |                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledLIN(<br/>    USART_TypeDef * USARTx)</code>                                                                              |
| Function description | Indicate if LIN mode is enabled.                                                                                                                                          |
| Parameters           | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                           |
| Return values        | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>Macro IS_UART_LIN_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.</li> </ul> |

Reference Manual to  
LL API cross  
reference:

- CR2 LINEN LL\_USART\_IsEnabledLIN

### **LL\_USART\_SetDEDeassertionTime**

|                                                   |                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_USART_SetDEDeassertionTime(USART_TypeDef * USARTx, uint32_t Time)</code></b>                                                                                   |
| Function description                              | Set DEDT (Driver Enable De-Assertion Time), Time value expressed on 5 bits ([4:0] bits).                                                                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Time:</b> Value between Min_Data=0 and Max_Data=31</li> </ul>                                              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                 |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_DRIVER_ENABLE_INSTANCE(USARTx) can be used to check whether or not Driver Enable feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 DEDT LL_USART_SetDEDeassertionTime</li> </ul>                                                                                                      |

### **LL\_USART\_GetDEDeassertionTime**

|                                                   |                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_USART_GetDEDeassertionTime(USART_TypeDef * USARTx)</code></b>                                                                                              |
| Function description                              | Return DEDT (Driver Enable De-Assertion Time)                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Time:</b> value expressed on 5 bits ([4:0] bits) : Value between Min_Data=0 and Max_Data=31</li> </ul>                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_DRIVER_ENABLE_INSTANCE(USARTx) can be used to check whether or not Driver Enable feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 DEDT LL_USART_GetDEDeassertionTime</li> </ul>                                                                                                      |

### **LL\_USART\_SetDEAssertionTime**

|                      |                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_USART_SetDEAssertionTime(USART_TypeDef * USARTx, uint32_t Time)</code></b>                                        |
| Function description | Set DEAT (Driver Enable Assertion Time), Time value expressed on 5 bits ([4:0] bits).                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Time:</b> Value between Min_Data=0 and Max_Data=31</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                    |

|                                             |                                                                                                                                                                                               |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_UART_DRIVER_ENABLE_INSTANCE(USARTx) can be used to check whether or not Driver Enable feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 DEAT LL_USART_SetDEAssertionTime</li> </ul>                                                                                                        |

### LL\_USART\_GetDEAssertionTime

|                                             |                                                                                                                                                                                               |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_USART_GetDEAssertionTime(<br/>    USART_TypeDef * USARTx)</code>                                                                                            |
| Function description                        | Return DEAT (Driver Enable Assertion Time)                                                                                                                                                    |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                                               |
| Return values                               | <ul style="list-style-type: none"> <li><b>Time:</b> value expressed on 5 bits ([4:0] bits) : Value between Min_Data=0 and Max_Data=31</li> </ul>                                              |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_UART_DRIVER_ENABLE_INSTANCE(USARTx) can be used to check whether or not Driver Enable feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 DEAT LL_USART_SetDEAssertionTime</li> </ul>                                                                                                        |

### LL\_USART\_EnableDEMode

|                                             |                                                                                                                                                                                               |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE void LL_USART_EnableDEMode(<br/>    USART_TypeDef * USARTx)</code>                                                                                                      |
| Function description                        | Enable Driver Enable (DE) Mode.                                                                                                                                                               |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                                               |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                 |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_UART_DRIVER_ENABLE_INSTANCE(USARTx) can be used to check whether or not Driver Enable feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR3 DEM LL_USART_EnableDEMode</li> </ul>                                                                                                               |

### LL\_USART\_DisableDEMode

|                      |                                                                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_USART_DisableDEMode(<br/>    USART_TypeDef * USARTx)</code>                                                                                                     |
| Function description | Disable Driver Enable (DE) Mode.                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                                               |
| Return values        | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                 |
| Notes                | <ul style="list-style-type: none"> <li>Macro IS_UART_DRIVER_ENABLE_INSTANCE(USARTx) can be used to check whether or not Driver Enable feature is supported by the USARTx instance.</li> </ul> |

Reference Manual to  
LL API cross  
reference:

- CR3 DEM LL\_USART\_DisableDEMode

### **LL\_USART\_IsEnabledDEMode**

|                                                   |                                                                                                                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledDEMode(<br/>    USART_TypeDef * USARTx)</code>                                                                                                 |
| Function description                              | Indicate if Driver Enable (DE) Mode is enabled.                                                                                                                                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_DRIVER_ENABLE_INSTANCE(USARTx) can be used to check whether or not Driver Enable feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 DEM LL_USART_IsEnabledDEMode</li> </ul>                                                                                                            |

### **LL\_USART\_SetDESignalPolarity**

|                                                   |                                                                                                                                                                                                                                                                                                                      |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_USART_SetDESignalPolarity(<br/>    USART_TypeDef * USARTx, uint32_t Polarity)</code>                                                                                                                                                                                                   |
| Function description                              | Select Driver Enable Polarity.                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Polarity:</b> This parameter can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_DE_POLARITY_HIGH</li> <li>– LL_USART_DE_POLARITY_LOW</li> </ul> </li> <li>• <b>None</b></li> </ul> |
| Return values                                     |                                                                                                                                                                                                                                                                                                                      |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_DRIVER_ENABLE_INSTANCE(USARTx) can be used to check whether or not Driver Enable feature is supported by the USARTx instance.</li> </ul>                                                                                                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 DEP LL_USART_SetDESignalPolarity</li> </ul>                                                                                                                                                                                                                             |

### **LL\_USART\_GetDESignalPolarity**

|                      |                                                                                                                                                                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_USART_GetDESignalPolarity(<br/>    USART_TypeDef * USARTx)</code>                                                                                                                                         |
| Function description | Return Driver Enable Polarity.                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                           |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Returned:</b> value can be one of the following values:           <ul style="list-style-type: none"> <li>– LL_USART_DE_POLARITY_HIGH</li> <li>– LL_USART_DE_POLARITY_LOW</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_UART_DRIVER_ENABLE_INSTANCE(USARTx)</li> </ul>                                                                                                                                            |

can be used to check whether or not Driver Enable feature is supported by the USARTx instance.

Reference Manual to  
LL API cross  
reference:

- CR3 DEP LL\_USART\_GetDESignalPolarity

### **LL\_USART\_ConfigAsyncMode**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_ConfigAsyncMode(<br/>USART_TypeDef * USARTx)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description                              | Perform basic configuration of USART for enabling use in Asynchronous Mode (UART)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Notes                                             | <ul style="list-style-type: none"> <li>• In UART mode, the following bits must be kept cleared: LINEN bit in the USART_CR2 register,CLKEN bit in the USART_CR2 register,SCEN bit in the USART_CR3 register,IREN bit in the USART_CR3 register,HDSEL bit in the USART_CR3 register.</li> <li>• Call of this function is equivalent to following function call sequence : Clear LINEN in CR2 using LL_USART_DisableLIN() functionClear CLKEN in CR2 using LL_USART_DisableSCLKOutput() functionClear SCEN in CR3 using LL_USART_DisableSmartcard() functionClear IREN in CR3 using LL_USART_DisableIrda() functionClear HDSEL in CR3 using LL_USART_DisableHalfDuplex() function</li> <li>• Other remaining configurations items related to Asynchronous Mode (as Baud Rate, Word length, Parity, ...) should be set using dedicated functions</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 LINEN LL_USART_ConfigAsyncMode</li> <li>• CR2 CLKEN LL_USART_ConfigAsyncMode</li> <li>• CR3 SCEN LL_USART_ConfigAsyncMode</li> <li>• CR3 IREN LL_USART_ConfigAsyncMode</li> <li>• CR3 HDSEL LL_USART_ConfigAsyncMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

### **LL\_USART\_ConfigSyncMode**

|                      |                                                                                                                                                                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_USART_ConfigSyncMode(<br/>USART_TypeDef * USARTx)</code></b>                                                                                                                                                                                                                            |
| Function description | Perform basic configuration of USART for enabling use in Synchronous Mode.                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                         |
| Notes                | <ul style="list-style-type: none"> <li>• In Synchronous mode, the following bits must be kept cleared: LINEN bit in the USART_CR2 register,SCEN bit in the USART_CR3 register,IREN bit in the USART_CR3 register,HDSEL bit in the USART_CR3 register. This function also sets the USART in Synchronous mode.</li> </ul> |

- Macro IS\_USART\_INSTANCE(USARTx) can be used to check whether or not Synchronous mode is supported by the USARTx instance.
- Call of this function is equivalent to following function call sequence : Clear LINEN in CR2 using LL\_USART\_DisableLIN() functionClear IREN in CR3 using LL\_USART\_DisableIrda() functionClear SCEN in CR3 using LL\_USART\_DisableSmartcard() functionClear HDSEL in CR3 using LL\_USART\_DisableHalfDuplex() functionSet CLKEN in CR2 using LL\_USART\_EnableSCLKOutput() function
- Other remaining configurations items related to Synchronous Mode (as Baud Rate, Word length, Parity, Clock Polarity, ...) should be set using dedicated functions

Reference Manual to  
LL API cross  
reference:

- CR2 LINEN LL\_USART\_ConfigSyncMode
- CR2 CLKEN LL\_USART\_ConfigSyncMode
- CR3 SCEN LL\_USART\_ConfigSyncMode
- CR3 IREN LL\_USART\_ConfigSyncMode
- CR3 HDSEL LL\_USART\_ConfigSyncMode

### **LL\_USART\_ConfigLINMode**

Function name

**`_STATIC_INLINE void LL_USART_ConfigLINMode(  
 USART_TypeDef * USARTx)`**

Function description

Perform basic configuration of USART for enabling use in LIN Mode.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Notes

- In LIN mode, the following bits must be kept cleared: STOP and CLKEN bits in the USART\_CR2 register,SCEN bit in the USART\_CR3 register,IREN bit in the USART\_CR3 register,HDSEL bit in the USART\_CR3 register. This function also set the UART/USART in LIN mode.
- Macro IS\_USART\_LIN\_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.
- Call of this function is equivalent to following function call sequence : Clear CLKEN in CR2 using LL\_USART\_DisableSCLKOutput() functionClear STOP in CR2 using LL\_USART\_SetStopBitsLength() functionClear SCEN in CR3 using LL\_USART\_DisableSmartcard() functionClear IREN in CR3 using LL\_USART\_DisableIrda() functionClear HDSEL in CR3 using LL\_USART\_DisableHalfDuplex() functionSet LINEN in CR2 using LL\_USART\_EnableLIN() function
- Other remaining configurations items related to LIN Mode (as Baud Rate, Word length, LIN Break Detection Length, ...) should be set using dedicated functions

Reference Manual to  
LL API cross  
reference:

- CR2 CLKEN LL\_USART\_ConfigLINMode
- CR2 STOP LL\_USART\_ConfigLINMode
- CR2 LINEN LL\_USART\_ConfigLINMode
- CR3 IREN LL\_USART\_ConfigLINMode

- CR3 SCEN LL\_USART\_ConfigLINMode
- CR3 HDSEL LL\_USART\_ConfigLINMode

### LL\_USART\_ConfigHalfDuplexMode

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_USART_ConfigHalfDuplexMode(USART_TypeDef * USARTx)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Function description                        | Perform basic configuration of USART for enabling use in Half Duplex Mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Notes                                       | <ul style="list-style-type: none"> <li>• In Half Duplex mode, the following bits must be kept cleared: LINEN bit in the USART_CR2 register,CLKEN bit in the USART_CR2 register,SCEN bit in the USART_CR3 register,IREN bit in the USART_CR3 register, This function also sets the USART/USART in Half Duplex mode.</li> <li>• Macro IS_UART_HALFDUPLEX_INSTANCE(USARTx) can be used to check whether or not Half-Duplex mode is supported by the USARTx instance.</li> <li>• Call of this function is equivalent to following function call sequence : Clear LINEN in CR2 using LL_USART_DisableLIN() functionClear CLKEN in CR2 using LL_USART_DisableSCLKOutput() functionClear SCEN in CR3 using LL_USART_DisableSmartcard() functionClear IREN in CR3 using LL_USART_DisableIrda() functionSet HDSEL in CR3 using LL_USART_EnableHalfDuplex() function</li> <li>• Other remaining configurations items related to Half Duplex Mode (as Baud Rate, Word length, Parity, ...) should be set using dedicated functions</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR2 LINEN LL_USART_ConfigHalfDuplexMode</li> <li>• CR2 CLKEN LL_USART_ConfigHalfDuplexMode</li> <li>• CR3 HDSEL LL_USART_ConfigHalfDuplexMode</li> <li>• CR3 SCEN LL_USART_ConfigHalfDuplexMode</li> <li>• CR3 IREN LL_USART_ConfigHalfDuplexMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

### LL\_USART\_ConfigSmartcardMode

|                      |                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_USART_ConfigSmartcardMode(USART_TypeDef * USARTx)</code></b>                                                                                                                                                                                                                                                          |
| Function description | Perform basic configuration of USART for enabling use in Smartcard Mode.                                                                                                                                                                                                                                                                              |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                                                                     |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                       |
| Notes                | <ul style="list-style-type: none"> <li>• In Smartcard mode, the following bits must be kept cleared: LINEN bit in the USART_CR2 register,IREN bit in the USART_CR3 register,HDSEL bit in the USART_CR3 register. This function also configures Stop bits to 1.5 bits and sets the USART in Smartcard mode (SCEN bit). Clock Output is also</li> </ul> |

enabled (CLKEN).

- Macro IS\_SMARTCARD\_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.
- Call of this function is equivalent to following function call sequence : Clear LINEN in CR2 using LL\_USART\_DisableLIN() functionClear IREN in CR3 using LL\_USART\_DisableIrda() functionClear HDSEL in CR3 using LL\_USART\_DisableHalfDuplex() functionConfigure STOP in CR2 using LL\_USART\_SetStopBitsLength() functionSet CLKEN in CR2 using LL\_USART\_EnableSCLKOutput() functionSet SCEN in CR3 using LL\_USART\_EnableSmartcard() function
- Other remaining configurations items related to Smartcard Mode (as Baud Rate, Word length, Parity, ...) should be set using dedicated functions

Reference Manual to  
LL API cross  
reference:

- CR2 LINEN LL\_USART\_ConfigSmartcardMode
- CR2 STOP LL\_USART\_ConfigSmartcardMode
- CR2 CLKEN LL\_USART\_ConfigSmartcardMode
- CR3 HDSEL LL\_USART\_ConfigSmartcardMode
- CR3 SCEN LL\_USART\_ConfigSmartcardMode

## LL\_USART\_ConfigIrdaMode

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>STATIC_INLINE void LL_USART_ConfigIrdaMode(<br/>USART_TypeDef * USARTx)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Function description | Perform basic configuration of USART for enabling use in Irda Mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• In IRDA mode, the following bits must be kept cleared: LINEN bit in the USART_CR2 register,STOP and CLKEN bits in the USART_CR2 register,SCEN bit in the USART_CR3 register,HDSEL bit in the USART_CR3 register. This function also sets the UART/USART in IRDA mode (IREN bit).</li> <li>• Macro IS_IRDA_INSTANCE(USARTx) can be used to check whether or not IrDA feature is supported by the USARTx instance.</li> <li>• Call of this function is equivalent to following function call sequence : Clear LINEN in CR2 using LL_USART_DisableLIN() functionClear CLKEN in CR2 using LL_USART_DisableSCLKOutput() functionClear SCEN in CR3 using LL_USART_DisableSmartcard() functionClear HDSEL in CR3 using LL_USART_DisableHalfDuplex() functionConfigure STOP in CR2 using LL_USART_SetStopBitsLength() functionSet IREN in CR3 using LL_USART_EnableIrda() function</li> <li>• Other remaining configurations items related to Irda Mode (as Baud Rate, Word length, Power mode, ...) should be set using dedicated functions</li> </ul> |
| Reference Manual to  | <ul style="list-style-type: none"> <li>• CR2 LINEN LL_USART_ConfigIrdaMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

- LL API cross reference:
- CR2 CLKEN LL\_USART\_ConfigIrdaMode
  - CR2 STOP LL\_USART\_ConfigIrdaMode
  - CR3 SCEN LL\_USART\_ConfigIrdaMode
  - CR3 HDSEL LL\_USART\_ConfigIrdaMode
  - CR3 IREN LL\_USART\_ConfigIrdaMode

### **LL\_USART\_ConfigMultiProcessMode**

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_USART_ConfigMultiProcessMode(<br/>USART_TypeDef * USARTx)</code></b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Function description                        | Perform basic configuration of USART for enabling use in Multi processor Mode (several USARTs connected in a network, one of the USARTs can be the master, its TX output connected to the RX inputs of the other slaves USARTs).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Notes                                       | <ul style="list-style-type: none"> <li>• In MultiProcessor mode, the following bits must be kept cleared: LINEN bit in the USART_CR2 register,CLKEN bit in the USART_CR2 register,SCEN bit in the USART_CR3 register,IREN bit in the USART_CR3 register,HDSEL bit in the USART_CR3 register.</li> <li>• Call of this function is equivalent to following function call sequence : Clear LINEN in CR2 using LL_USART_DisableLIN() functionClear CLKEN in CR2 using LL_USART_DisableSCLKOutput() functionClear SCEN in CR3 using LL_USART_DisableSmartcard() functionClear IREN in CR3 using LL_USART_DisableIrda() functionClear HDSEL in CR3 using LL_USART_DisableHalfDuplex() function</li> <li>• Other remaining configurations items related to Multi processor Mode (as Baud Rate, Wake Up Method, Node address, ...) should be set using dedicated functions</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• CR2 LINEN LL_USART_ConfigMultiProcessMode</li> <li>• CR2 CLKEN LL_USART_ConfigMultiProcessMode</li> <li>• CR3 SCEN LL_USART_ConfigMultiProcessMode</li> <li>• CR3 HDSEL LL_USART_ConfigMultiProcessMode</li> <li>• CR3 IREN LL_USART_ConfigMultiProcessMode</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

### **LL\_USART\_IsActiveFlag\_PE**

|                                             |                                                                                                   |
|---------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_USART_IsActiveFlag_PE(<br/>USART_TypeDef * USARTx)</code></b> |
| Function description                        | Check if the USART Parity Error Flag is set or not.                                               |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                 |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• ISR PE LL_USART_IsActiveFlag_PE</li> </ul>               |

**LL\_USART\_IsActiveFlag\_FE**

Function name **\_\_STATIC\_INLINE uint32\_t LL\_USART\_IsActiveFlag\_FE  
(USART\_TypeDef \* USARTx)**

Function description Check if the USART Framing Error Flag is set or not.

Parameters • **USARTx:** USART Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
• ISR FE LL\_USART\_IsActiveFlag\_FE

**LL\_USART\_IsActiveFlag\_NE**

Function name **\_\_STATIC\_INLINE uint32\_t LL\_USART\_IsActiveFlag\_NE  
(USART\_TypeDef \* USARTx)**

Function description Check if the USART Noise error detected Flag is set or not.

Parameters • **USARTx:** USART Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
• ISR NF LL\_USART\_IsActiveFlag\_NE

**LL\_USART\_IsActiveFlag\_ORE**

Function name **\_\_STATIC\_INLINE uint32\_t LL\_USART\_IsActiveFlag\_ORE  
(USART\_TypeDef \* USARTx)**

Function description Check if the USART OverRun Error Flag is set or not.

Parameters • **USARTx:** USART Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
• ISR ORE LL\_USART\_IsActiveFlag\_ORE

**LL\_USART\_IsActiveFlag\_IDLE**

Function name **\_\_STATIC\_INLINE uint32\_t LL\_USART\_IsActiveFlag\_IDLE  
(USART\_TypeDef \* USARTx)**

Function description Check if the USART IDLE line detected Flag is set or not.

Parameters • **USARTx:** USART Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
• ISR IDLE LL\_USART\_IsActiveFlag\_IDLE

**LL\_USART\_IsActiveFlag\_RXNE**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_RXNE(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Read Data Register Not Empty Flag is set or not.                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTTx:</b> USART Instance</li> </ul>                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR RXNE LL_USART_IsActiveFlag_RXNE</li> </ul>           |

**LL\_USART\_IsActiveFlag\_TC**

|                                                   |                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_TC(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Transmission Complete Flag is set or not.                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTTx:</b> USART Instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR TC LL_USART_IsActiveFlag_TC</li> </ul>             |

**LL\_USART\_IsActiveFlag\_TXE**

|                                                   |                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_TXE(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Transmit Data Register Empty Flag is set or not.                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTTx:</b> USART Instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR TXE LL_USART_IsActiveFlag_TXE</li> </ul>            |

**LL\_USART\_IsActiveFlag\_LBD**

|                      |                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_LBD(<br/>    USART_TypeDef * USARTx)</code>                                                                            |
| Function description | Check if the USART LIN Break Detection Flag is set or not.                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTTx:</b> USART Instance</li> </ul>                                                                                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                          |
| Notes                | <ul style="list-style-type: none"> <li>• Macro IS_UART_LIN_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.</li> </ul> |

- Reference Manual to  
LL API cross  
reference:
- ISR LBDF LL\_USART\_IsActiveFlag\_LBD

### **LL\_USART\_IsActiveFlag\_nCTS**

|                                                   |                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_USART_IsActiveFlag_nCTS(<br/>USART_TypeDef * USARTx)</code></b>                                                                                              |
| Function description                              | Check if the USART CTS interrupt Flag is set or not.                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_HWFLOW_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR CTSIF LL_USART_IsActiveFlag_nCTS</li> </ul>                                                                                                         |

### **LL\_USART\_IsActiveFlag\_CTS**

|                                                   |                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE uint32_t LL_USART_IsActiveFlag_CTS(<br/>USART_TypeDef * USARTx)</code></b>                                                                                               |
| Function description                              | Check if the USART CTS Flag is set or not.                                                                                                                                                       |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                               |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_HWFLOW_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR CTS LL_USART_IsActiveFlag_CTS</li> </ul>                                                                                                            |

### **LL\_USART\_IsActiveFlag\_RTO**

|                      |                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE uint32_t LL_USART_IsActiveFlag_RTO(<br/>USART_TypeDef * USARTx)</code></b> |
| Function description | Check if the USART Receiver Time Out Flag is set or not.                                           |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                 |

Reference Manual to  
LL API cross  
reference:

**LL\_USART\_IsActiveFlag\_EOB**

|                                             |                                                                                                                                                                                                 |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_USART_IsActiveFlag_EOB(USART_TypeDef * USARTx)</code></b>                                                                                                   |
| Function description                        | Check if the USART End Of Block Flag is set or not.                                                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                               |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                              |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro <code>IS_SMARTCARD_INSTANCE(USARTx)</code> can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• ISR EOBF <code>LL_USART_IsActiveFlag_EOB</code></li> </ul>                                                                                             |

**LL\_USART\_IsActiveFlag\_ABRE**

|                                             |                                                                                                                                                                                                                                   |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_USART_IsActiveFlag_ABRE(USART_TypeDef * USARTx)</code></b>                                                                                                                                    |
| Function description                        | Check if the USART Auto-Baud Rate Error Flag is set or not.                                                                                                                                                                       |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                 |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro <code>IS_USART_AUTOBAUDRATE_DETECTION_INSTANCE(USARTx)</code> can be used to check whether or not Auto Baud Rate detection feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• ISR ABRE <code>LL_USART_IsActiveFlag_ABRE</code></li> </ul>                                                                                                                              |

**LL\_USART\_IsActiveFlag\_ABR**

|                                             |                                                                                                                                                                                                                                   |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_USART_IsActiveFlag_ABR(USART_TypeDef * USARTx)</code></b>                                                                                                                                     |
| Function description                        | Check if the USART Auto-Baud Rate Flag is set or not.                                                                                                                                                                             |
| Parameters                                  | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                 |
| Return values                               | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                                |
| Notes                                       | <ul style="list-style-type: none"> <li>• Macro <code>IS_USART_AUTOBAUDRATE_DETECTION_INSTANCE(USARTx)</code> can be used to check whether or not Auto Baud Rate detection feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>• ISR ABRF <code>LL_USART_IsActiveFlag_ABR</code></li> </ul>                                                                                                                               |

**LL\_USART\_IsActiveFlag\_BUSY**

|                                                   |                                                                                               |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_BUSY<br/>(USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Busy Flag is set or not.                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>             |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>            |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR BUSY LL_USART_IsActiveFlag_BUSY</li> </ul>       |

**LL\_USART\_IsActiveFlag\_CM**

|                                                   |                                                                                             |
|---------------------------------------------------|---------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_CM<br/>(USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Character Match Flag is set or not.                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR CMF LL_USART_IsActiveFlag_CM</li> </ul>        |

**LL\_USART\_IsActiveFlag\_SBK**

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_SBK<br/>(USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Send Break Flag is set or not.                                            |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR SBKF LL_USART_IsActiveFlag_SBK</li> </ul>       |

**LL\_USART\_IsActiveFlag\_RWU**

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_RWU<br/>(USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Receive Wake Up from mute mode Flag is set or not.                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR RWU LL_USART_IsActiveFlag_RWU</li> </ul>        |

**LL\_USART\_IsActiveFlag\_WKUP**

|                                                   |                                                                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_WKUP(<br/>    USART_TypeDef * USARTx)</code>                                                                                                                           |
| Function description                              | Check if the USART Wake Up from stop mode Flag is set or not.                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                                          |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro<br/><code>IS_UART_WAKEUP_FROMSTOP_INSTANCE(USARTx)</code> can be used to check whether or not Wake-up from Stop mode feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR WUF LL_USART_IsActiveFlag_WKUP</li> </ul>                                                                                                                                      |

**LL\_USART\_IsActiveFlag\_TEACK**

|                                                   |                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_TEACK(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Transmit Enable Acknowledge Flag is set or not.                                 |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR TEACK LL_USART_IsActiveFlag_TEACK</li> </ul>          |

**LL\_USART\_IsActiveFlag\_REACK**

|                                                   |                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsActiveFlag_REACK(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Receive Enable Acknowledge Flag is set or not.                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                 |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• ISR REACK LL_USART_IsActiveFlag_REACK</li> </ul>          |

**LL\_USART\_ClearFlag\_PE**

|                      |                                                                                          |
|----------------------|------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_USART_ClearFlag_PE(<br/>    USART_TypeDef * USARTx)</code> |
| Function description | Clear Parity Error Flag.                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                          |

Reference Manual to  
LL API cross  
reference:

- ICR PECF LL\_USART\_ClearFlag\_PE

### **LL\_USART\_ClearFlag\_FE**

Function name

**`_STATIC_INLINE void LL_USART_ClearFlag_FE  
(USART_TypeDef * USARTx)`**

Function description

Clear Framing Error Flag.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Reference Manual to

LL API cross

reference:

- ICR FECF LL\_USART\_ClearFlag\_FE

### **LL\_USART\_ClearFlag\_NE**

Function name

**`_STATIC_INLINE void LL_USART_ClearFlag_NE  
(USART_TypeDef * USARTx)`**

Function description

Clear Noise detected Flag.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Reference Manual to

LL API cross

reference:

- ICR NCF LL\_USART\_ClearFlag\_NE

### **LL\_USART\_ClearFlag\_ORE**

Function name

**`_STATIC_INLINE void LL_USART_ClearFlag_ORE  
(USART_TypeDef * USARTx)`**

Function description

Clear OverRun Error Flag.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Reference Manual to

LL API cross

reference:

- ICR ORECF LL\_USART\_ClearFlag\_ORE

### **LL\_USART\_ClearFlag\_IDLE**

Function name

**`_STATIC_INLINE void LL_USART_ClearFlag_IDLE  
(USART_TypeDef * USARTx)`**

Function description

Clear IDLE line detected Flag.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

- Reference Manual to  
LL API cross  
reference:
- ICR IDLECF LL\_USART\_ClearFlag\_IDLE

### LL\_USART\_ClearFlag\_TC

Function name **`_STATIC_INLINE void LL_USART_ClearFlag_TC  
(USART_TypeDef * USARTx)`**

Function description Clear Transmission Complete Flag.

- Parameters
- **USARTx**: USART Instance

- Return values
- **None**

Reference Manual to  
LL API cross  
reference:

- ICR TCCF LL\_USART\_ClearFlag\_TC

### LL\_USART\_ClearFlag\_LBD

Function name **`_STATIC_INLINE void LL_USART_ClearFlag_LBD  
(USART_TypeDef * USARTx)`**

Function description Clear LIN Break Detection Flag.

- Parameters
- **USARTx**: USART Instance

- Return values
- **None**

Notes

- Macro IS\_UART\_LIN\_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.

Reference Manual to  
LL API cross  
reference:

- ICR LBDCF LL\_USART\_ClearFlag\_LBD

### LL\_USART\_ClearFlag\_nCTS

Function name **`_STATIC_INLINE void LL_USART_ClearFlag_nCTS  
(USART_TypeDef * USARTx)`**

Function description Clear CTS Interrupt Flag.

- Parameters
- **USARTx**: USART Instance

- Return values
- **None**

Notes

- Macro IS\_UART\_HWFLOW\_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.

Reference Manual to  
LL API cross  
reference:

- ICR CTSCF LL\_USART\_ClearFlag\_nCTS

**LL\_USART\_ClearFlag\_RTO**

Function name **`_STATIC_INLINE void LL_USART_ClearFlag_RTO(USART_TypeDef * USARTx)`**

Function description Clear Receiver Time Out Flag.

Parameters • **USARTx:** USART Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
• ICR RTOCF LL\_USART\_ClearFlag\_RTO

**LL\_USART\_ClearFlag\_EOB**

Function name **`_STATIC_INLINE void LL_USART_ClearFlag_EOB(USART_TypeDef * USARTx)`**

Function description Clear End Of Block Flag.

Parameters • **USARTx:** USART Instance

Return values • **None**

Notes • Macro IS\_SMARTCARD\_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.

Reference Manual to  
LL API cross  
reference:  
• ICR EOBCF LL\_USART\_ClearFlag\_EOB

**LL\_USART\_ClearFlag\_CM**

Function name **`_STATIC_INLINE void LL_USART_ClearFlag_CM(USART_TypeDef * USARTx)`**

Function description Clear Character Match Flag.

Parameters • **USARTx:** USART Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:  
• ICR CMCF LL\_USART\_ClearFlag\_CM

**LL\_USART\_ClearFlag\_WKUP**

Function name **`_STATIC_INLINE void LL_USART_ClearFlag_WKUP(USART_TypeDef * USARTx)`**

Function description Clear Wake Up from stop mode Flag.

Parameters • **USARTx:** USART Instance

Return values • **None**

Notes • Macro  
IS\_UART\_WAKEUP\_FROMSTOP\_INSTANCE(USARTx) can be used to check whether or not Wake-up from Stop mode

feature is supported by the USARTx instance.

Reference Manual to  
LL API cross  
reference:

- ICR WUCF LL\_USART\_ClearFlag\_WKUP

### LL\_USART\_EnableIT\_IDLE

Function name

`__STATIC_INLINE void LL_USART_EnableIT_IDLE  
(USART_TypeDef * USARTx)`

Function description

Enable IDLE Interrupt.

Parameters

- **USARTx**: USART Instance

Return values

- **None**

Reference Manual to

LL API cross

reference:

- CR1 IDLEIE LL\_USART\_EnableIT\_IDLE

### LL\_USART\_EnableIT\_RXNE

Function name

`__STATIC_INLINE void LL_USART_EnableIT_RXNE  
(USART_TypeDef * USARTx)`

Function description

Enable RX Not Empty Interrupt.

Parameters

- **USARTx**: USART Instance

Return values

- **None**

Reference Manual to

LL API cross

reference:

- CR1 RXNEIE LL\_USART\_EnableIT\_RXNE

### LL\_USART\_EnableIT\_TC

Function name

`__STATIC_INLINE void LL_USART_EnableIT_TC  
(USART_TypeDef * USARTx)`

Function description

Enable Transmission Complete Interrupt.

Parameters

- **USARTx**: USART Instance

Return values

- **None**

Reference Manual to

LL API cross

reference:

- CR1 TCIE LL\_USART\_EnableIT\_TC

### LL\_USART\_EnableIT\_TXE

Function name

`__STATIC_INLINE void LL_USART_EnableIT_TXE  
(USART_TypeDef * USARTx)`

Function description

Enable TX Empty Interrupt.

Parameters

- **USARTx**: USART Instance

Return values

- **None**

Reference Manual to  
LL API cross  
reference:

- CR1 TXEIE LL\_USART\_EnableIT\_TXE

### **LL\_USART\_EnableIT\_PE**

Function name

**`_STATIC_INLINE void LL_USART_EnableIT_PE  
(USART_TypeDef * USARTx)`**

Function description

Enable Parity Error Interrupt.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Reference Manual to

LL API cross

reference:

- CR1 PEIE LL\_USART\_EnableIT\_PE

### **LL\_USART\_EnableIT\_CM**

Function name

**`_STATIC_INLINE void LL_USART_EnableIT_CM  
(USART_TypeDef * USARTx)`**

Function description

Enable Character Match Interrupt.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Reference Manual to

LL API cross

reference:

- CR1 CMIE LL\_USART\_EnableIT\_CM

### **LL\_USART\_EnableIT\_RTO**

Function name

**`_STATIC_INLINE void LL_USART_EnableIT_RTO  
(USART_TypeDef * USARTx)`**

Function description

Enable Receiver Timeout Interrupt.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Reference Manual to

LL API cross

reference:

- CR1 RTOIE LL\_USART\_EnableIT\_RTO

### **LL\_USART\_EnableIT\_EOB**

Function name

**`_STATIC_INLINE void LL_USART_EnableIT_EOB  
(USART_TypeDef * USARTx)`**

Function description

Enable End Of Block Interrupt.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Notes

- Macro IS\_SMARTCARD\_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the

USARTx instance.

Reference Manual to  
LL API cross  
reference:

- CR1 EOBIIE LL\_USART\_EnableIT\_EOB

### **LL\_USART\_EnableIT\_LBD**

Function name

**`_STATIC_INLINE void LL_USART_EnableIT_LBD  
(USART_TypeDef * USARTx)`**

Function description

Enable LIN Break Detection Interrupt.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Notes

- Macro IS\_UART\_LIN\_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.

Reference Manual to  
LL API cross  
reference:

- CR2 LBDIE LL\_USART\_EnableIT\_LBD

### **LL\_USART\_EnableIT\_ERROR**

Function name

**`_STATIC_INLINE void LL_USART_EnableIT_ERROR  
(USART_TypeDef * USARTx)`**

Function description

Enable Error Interrupt.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Notes

- When set, Error Interrupt Enable Bit is enabling interrupt generation in case of a framing error, overrun error or noise flag (FE=1 or ORE=1 or NF=1 in the USARTx\_ISR register). 0: Interrupt is inhibited 1: An interrupt is generated when FE=1 or ORE=1 or NF=1 in the USARTx\_ISR register.

Reference Manual to  
LL API cross  
reference:

- CR3 EIE LL\_USART\_EnableIT\_ERROR

### **LL\_USART\_EnableIT\_CTS**

Function name

**`_STATIC_INLINE void LL_USART_EnableIT_CTS  
(USART_TypeDef * USARTx)`**

Function description

Enable CTS Interrupt.

Parameters

- **USARTx:** USART Instance

Return values

- **None**

Notes

- Macro IS\_UART\_HWFLOW\_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.

- Reference Manual to  
LL API cross  
reference:
- CR3 CTSIE LL\_USART\_EnableIT\_CTS

### **LL\_USART\_EnableIT\_WKUP**

|                                                   |                                                                                                                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_EnableIT_WKUP(<br/>USART_TypeDef * USARTx)</code></b>                                                                                                                |
| Function description                              | Enable Wake Up from Stop Mode Interrupt.                                                                                                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                          |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                            |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_WAKEUP_FROMSTOP_INSTANCE(USARTx) can be used to check whether or not Wake-up from Stop mode feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR3 WUFIE LL_USART_EnableIT_WKUP</li> </ul>                                                                                                                       |

### **LL\_USART\_DisableIT\_IDLE**

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_DisableIT_IDLE(<br/>USART_TypeDef * USARTx)</code></b> |
| Function description                              | Disable IDLE Interrupt.                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 IDLEIE LL_USART_DisableIT_IDLE</li> </ul>       |

### **LL\_USART\_DisableIT\_RXNE**

|                                                   |                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_DisableIT_RXNE(<br/>USART_TypeDef * USARTx)</code></b> |
| Function description                              | Disable RX Not Empty Interrupt.                                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>            |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 RXNEIE LL_USART_DisableIT_RXNE</li> </ul>       |

### **LL\_USART\_DisableIT\_TC**

|                      |                                                                                            |
|----------------------|--------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_USART_DisableIT_TC(<br/>USART_TypeDef * USARTx)</code></b> |
| Function description | Disable Transmission Complete Interrupt.                                                   |

---

|                                                   |                                                                                  |
|---------------------------------------------------|----------------------------------------------------------------------------------|
| Parameters                                        | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>  |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 TCIE LL_USART_DisableIT_TC</li> </ul> |

### LL\_USART\_DisableIT\_TXE

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_USART_DisableIT_TXE(USART_TypeDef * USARTx)</code></b> |
| Function description                              | Disable TX Empty Interrupt.                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 TXEIE LL_USART_DisableIT_TXE</li> </ul>      |

### LL\_USART\_DisableIT\_PE

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_USART_DisableIT_PE(USART_TypeDef * USARTx)</code></b> |
| Function description                              | Disable Parity Error Interrupt.                                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 PEIE LL_USART_DisableIT_PE</li> </ul>       |

### LL\_USART\_DisableIT\_CM

|                                                   |                                                                                        |
|---------------------------------------------------|----------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_USART_DisableIT_CM(USART_TypeDef * USARTx)</code></b> |
| Function description                              | Disable Character Match Interrupt.                                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>        |
| Return values                                     | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                          |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>CR1 CMIE LL_USART_DisableIT_CM</li> </ul>       |

### LL\_USART\_DisableIT\_RTO

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_USART_DisableIT_RTO(USART_TypeDef * USARTx)</code></b> |
| Function description | Disable Receiver Timeout Interrupt.                                                     |
| Parameters           | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>         |

---

|                                                   |                                                                                      |
|---------------------------------------------------|--------------------------------------------------------------------------------------|
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 RTOIE LL_USART_DisableIT_RTO</li> </ul> |

### LL\_USART\_DisableIT\_EOB

|                                                   |                                                                                                                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_DisableIT_EOB<br/>(USART_TypeDef * USARTx)</code></b>                                                                                        |
| Function description                              | Disable End Of Block Interrupt.                                                                                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                    |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 EOBIIE LL_USART_DisableIT_EOB</li> </ul>                                                                                              |

### LL\_USART\_DisableIT\_LBD

|                                                   |                                                                                                                                                                             |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_DisableIT_LBD<br/>(USART_TypeDef * USARTx)</code></b>                                                                                 |
| Function description                              | Disable LIN Break Detection Interrupt.                                                                                                                                      |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                           |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                             |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro IS_UART_LIN_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR2 LBDIE LL_USART_DisableIT_LBD</li> </ul>                                                                                        |

### LL\_USART\_DisableIT\_ERROR

|                      |                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_USART_DisableIT_ERROR<br/>(USART_TypeDef * USARTx)</code></b>                                                                                                                                                                                                                                                       |
| Function description | Disable Error Interrupt.                                                                                                                                                                                                                                                                                                                            |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                                                                                                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                     |
| Notes                | <ul style="list-style-type: none"> <li>• When set, Error Interrupt Enable Bit is enabling interrupt generation in case of a framing error, overrun error or noise flag (FE=1 or ORE=1 or NF=1 in the USARTx_ISR register). 0: Interrupt is inhibited 1: An interrupt is generated when FE=1 or ORE=1 or NF=1 in the USARTx_ISR register.</li> </ul> |

- Reference Manual to LL API cross reference:
- CR3 EIE LL\_USART\_DisableIT\_ERROR

### **LL\_USART\_DisableIT\_CTS**

|                                             |                                                                                                                                                                                                |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_USART_DisableIT_CTS(<br/>USART_TypeDef * USARTx)</code></b>                                                                                                    |
| Function description                        | Disable CTS Interrupt.                                                                                                                                                                         |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                                                |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                  |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_UART_HWFLOW_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR3 CTSIE LL_USART_DisableIT_CTS</li> </ul>                                                                                                             |

### **LL\_USART\_DisableIT\_WKUP**

|                                             |                                                                                                                                                                                                          |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE void LL_USART_DisableIT_WKUP(<br/>USART_TypeDef * USARTx)</code></b>                                                                                                             |
| Function description                        | Disable Wake Up from Stop Mode Interrupt.                                                                                                                                                                |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                                                          |
| Return values                               | <ul style="list-style-type: none"> <li><b>None</b></li> </ul>                                                                                                                                            |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_UART_WAKEUP_FROMSTOP_INSTANCE(USARTx) can be used to check whether or not Wake-up from Stop mode feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR3 WUFIE LL_USART_DisableIT_WKUP</li> </ul>                                                                                                                      |

### **LL\_USART\_IsEnabledIT\_IDLE**

|                                             |                                                                                                    |
|---------------------------------------------|----------------------------------------------------------------------------------------------------|
| Function name                               | <b><code>_STATIC_INLINE uint32_t LL_USART_IsEnabledIT_IDLE(<br/>USART_TypeDef * USARTx)</code></b> |
| Function description                        | Check if the USART IDLE Interrupt source is enabled or disabled.                                   |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                    |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                   |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR1 IDLEIE LL_USART_IsEnabledIT_IDLE</li> </ul>             |

**LL\_USART\_IsEnabledIT\_RXNE**

|                                                   |                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledIT_RXNE(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART RX Not Empty Interrupt is enabled or disabled.                                |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 RXNEIE LL_USART_IsEnabledIT_RXNE</li> </ul>         |

**LL\_USART\_IsEnabledIT\_TC**

|                                                   |                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledIT_TC(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Transmission Complete Interrupt is enabled or disabled.                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 TCIE LL_USART_IsEnabledIT_TC</li> </ul>           |

**LL\_USART\_IsEnabledIT\_TXE**

|                                                   |                                                                                                 |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledIT_TXE(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART TX Empty Interrupt is enabled or disabled.                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>               |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>              |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 TXEIE LL_USART_IsEnabledIT_TXE</li> </ul>          |

**LL\_USART\_IsEnabledIT\_PE**

|                                                   |                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledIT_PE(<br/>    USART_TypeDef * USARTx)</code> |
| Function description                              | Check if the USART Parity Error Interrupt is enabled or disabled.                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>              |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>             |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR1 PEIE LL_USART_IsEnabledIT_PE</li> </ul>           |

**LL\_USART\_IsEnabledIT\_CM**

Function name **`_STATIC_INLINE uint32_t LL_USART_IsEnabledIT_CM  
(USART_TypeDef * USARTx)`**

Function description Check if the USART Character Match Interrupt is enabled or disabled.

Parameters • **USARTx:** USART Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
CR1 CMIE LL\_USART\_IsEnabledIT\_CM

**LL\_USART\_IsEnabledIT\_RTO**

Function name **`_STATIC_INLINE uint32_t LL_USART_IsEnabledIT_RTO  
(USART_TypeDef * USARTx)`**

Function description Check if the USART Receiver Timeout Interrupt is enabled or disabled.

Parameters • **USARTx:** USART Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
CR1 RTOIE LL\_USART\_IsEnabledIT\_RTO

**LL\_USART\_IsEnabledIT\_EOB**

Function name **`_STATIC_INLINE uint32_t LL_USART_IsEnabledIT_EOB  
(USART_TypeDef * USARTx)`**

Function description Check if the USART End Of Block Interrupt is enabled or disabled.

Parameters • **USARTx:** USART Instance

Return values • **State:** of bit (1 or 0).

Notes • Macro IS\_SMARTCARD\_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.

Reference Manual to  
LL API cross  
reference:  
CR1 EOBIIE LL\_USART\_IsEnabledIT\_EOB

**LL\_USART\_IsEnabledIT\_LBD**

Function name **`_STATIC_INLINE uint32_t LL_USART_IsEnabledIT_LBD  
(USART_TypeDef * USARTx)`**

Function description Check if the USART LIN Break Detection Interrupt is enabled or disabled.

Parameters • **USARTx:** USART Instance

Return values • **State:** of bit (1 or 0).

---

|                                             |                                                                                                                                                                           |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_UART_LIN_INSTANCE(USARTx) can be used to check whether or not LIN feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR2 LBDIE LL_USART_IsEnabledIT_LBD</li> </ul>                                                                                      |

### LL\_USART\_IsEnabledIT\_ERROR

|                                             |                                                                                                       |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledIT_ERROR(<br/>        USART_TypeDef * USARTx)</code> |
| Function description                        | Check if the USART Error Interrupt is enabled or disabled.                                            |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                       |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                      |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR3 EIE LL_USART_IsEnabledIT_ERROR</li> </ul>                  |

### LL\_USART\_IsEnabledIT\_CTS

|                                             |                                                                                                                                                                                                |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                               | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledIT_CTS(<br/>        USART_TypeDef * USARTx)</code>                                                                                            |
| Function description                        | Check if the USART CTS Interrupt is enabled or disabled.                                                                                                                                       |
| Parameters                                  | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                                                |
| Return values                               | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                               |
| Notes                                       | <ul style="list-style-type: none"> <li>Macro IS_UART_HWFLOW_INSTANCE(USARTx) can be used to check whether or not Hardware Flow control feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross reference: | <ul style="list-style-type: none"> <li>CR3 CTSIE LL_USART_IsEnabledIT_CTS</li> </ul>                                                                                                           |

### LL\_USART\_IsEnabledIT\_WKUP

|                                  |                                                                                                                                                                                                          |
|----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                    | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledIT_WKUP(<br/>        USART_TypeDef * USARTx)</code>                                                                                                     |
| Function description             | Check if the USART Wake Up from Stop Mode Interrupt is enabled or disabled.                                                                                                                              |
| Parameters                       | <ul style="list-style-type: none"> <li><b>USARTx:</b> USART Instance</li> </ul>                                                                                                                          |
| Return values                    | <ul style="list-style-type: none"> <li><b>State:</b> of bit (1 or 0).</li> </ul>                                                                                                                         |
| Notes                            | <ul style="list-style-type: none"> <li>Macro IS_UART_WAKEUP_FROMSTOP_INSTANCE(USARTx) can be used to check whether or not Wake-up from Stop mode feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual to LL API cross | <ul style="list-style-type: none"> <li>CR3 WUFIE LL_USART_IsEnabledIT_WKUP</li> </ul>                                                                                                                    |

reference:

### LL\_USART\_EnableDMAReq\_RX

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_USART_EnableDMAReq_RX(USART_TypeDef * USARTx)</code></b> |
| Function description                              | Enable DMA Mode for reception.                                                            |
| Parameters                                        | <ul style="list-style-type: none"><li><b>USARTx</b>: USART Instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li><b>None</b></li></ul>                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR3 DMAR LL_USART_EnableDMAReq_RX</li></ul>         |

### LL\_USART\_DisableDMAReq\_RX

|                                                   |                                                                                            |
|---------------------------------------------------|--------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_USART_DisableDMAReq_RX(USART_TypeDef * USARTx)</code></b> |
| Function description                              | Disable DMA Mode for reception.                                                            |
| Parameters                                        | <ul style="list-style-type: none"><li><b>USARTx</b>: USART Instance</li></ul>              |
| Return values                                     | <ul style="list-style-type: none"><li><b>None</b></li></ul>                                |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR3 DMAR LL_USART_DisableDMAReq_RX</li></ul>         |

### LL\_USART\_IsEnabledDMAReq\_RX

|                                                   |                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE uint32_t LL_USART_IsEnabledDMAReq_RX(USART_TypeDef * USARTx)</code></b> |
| Function description                              | Check if DMA Mode is enabled for reception.                                                      |
| Parameters                                        | <ul style="list-style-type: none"><li><b>USARTx</b>: USART Instance</li></ul>                    |
| Return values                                     | <ul style="list-style-type: none"><li><b>State</b>: of bit (1 or 0).</li></ul>                   |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR3 DMAR LL_USART_IsEnabledDMAReq_RX</li></ul>             |

### LL\_USART\_EnableDMAReq\_TX

|                                                   |                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>__STATIC_INLINE void LL_USART_EnableDMAReq_TX(USART_TypeDef * USARTx)</code></b> |
| Function description                              | Enable DMA Mode for transmission.                                                         |
| Parameters                                        | <ul style="list-style-type: none"><li><b>USARTx</b>: USART Instance</li></ul>             |
| Return values                                     | <ul style="list-style-type: none"><li><b>None</b></li></ul>                               |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>CR3 DMAT LL_USART_EnableDMAReq_TX</li></ul>         |

**LL\_USART\_DisableDMAReq\_TX**

Function name      **STATIC\_INLINE void LL\_USART\_DisableDMAReq\_TX  
(USART\_TypeDef \* USARTx)**

Function description      Disable DMA Mode for transmission.

Parameters      • **USARTx:** USART Instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR3 DMAT LL\_USART\_DisableDMAReq\_TX

**LL\_USART\_IsEnabledDMAReq\_TX**

Function name      **STATIC\_INLINE uint32\_t LL\_USART\_IsEnabledDMAReq\_TX  
(USART\_TypeDef \* USARTx)**

Function description      Check if DMA Mode is enabled for transmission.

Parameters      • **USARTx:** USART Instance

Return values      • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:  
• CR3 DMAT LL\_USART\_IsEnabledDMAReq\_TX

**LL\_USART\_EnableDMADeactOnRxErr**

Function name      **STATIC\_INLINE void LL\_USART\_EnableDMADeactOnRxErr  
(USART\_TypeDef \* USARTx)**

Function description      Enable DMA Disabling on Reception Error.

Parameters      • **USARTx:** USART Instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR3 DDRE LL\_USART\_EnableDMADeactOnRxErr

**LL\_USART\_DisableDMADeactOnRxErr**

Function name      **STATIC\_INLINE void LL\_USART\_DisableDMADeactOnRxErr  
(USART\_TypeDef \* USARTx)**

Function description      Disable DMA Disabling on Reception Error.

Parameters      • **USARTx:** USART Instance

Return values      • **None**

Reference Manual to  
LL API cross  
reference:  
• CR3 DDRE LL\_USART\_DisableDMADeactOnRxErr

**LL\_USART\_IsEnabledDMADeactOnRxErr**

|                                                   |                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_IsEnabledDMADeactOnRxErr (USART_TypeDef * USARTx)</code> |
| Function description                              | Indicate if DMA Disabling on Reception Error is disabled.                                        |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>               |
| Reference Manual to<br>LL API cross<br>reference: | CR3 DDRE LL_USART_IsEnabledDMADeactOnRxErr                                                       |

**LL\_USART\_DMA\_GetRegAddr**

|                                                   |                                                                                                                                                                                                                                                                                                |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_USART_DMA_GetRegAddr (USART_TypeDef * USARTx, uint32_t Direction)</code>                                                                                                                                                                                     |
| Function description                              | Get the data register address used for DMA transfer.                                                                                                                                                                                                                                           |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Direction:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>- LL_USART_DMA_REG_DATA_TRANSMIT</li> <li>- LL_USART_DMA_REG_DATA_RECEIVE</li> </ul> </li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Address:</b> of data register</li> </ul>                                                                                                                                                                                                           |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• RDR RDR LL_USART_DMA_GetRegAddr</li> <li>• TDR TDR LL_USART_DMA_GetRegAddr</li> </ul>                                                                                                                                                                 |

**LL\_USART\_ReceiveData8**

|                                                   |                                                                                                           |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint8_t LL_USART_ReceiveData8 (USART_TypeDef * USARTx)</code>                       |
| Function description                              | Read Receiver Data register (Receive Data value, 8 bits)                                                  |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xFF</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | • RDR RDR LL_USART_ReceiveData8                                                                           |

**LL\_USART\_ReceiveData9**

|                      |                                                                                                            |
|----------------------|------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint16_t LL_USART_ReceiveData9 (USART_TypeDef * USARTx)</code>                       |
| Function description | Read Receiver Data register (Receive Data value, 9 bits)                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                          |
| Return values        | <ul style="list-style-type: none"> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x1FF</li> </ul> |
| Reference Manual to  | • RDR RDR LL_USART_ReceiveData9                                                                            |

LL API cross  
reference:

### **LL\_USART\_TransmitData8**

|                                                   |                                                                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_TransmitData8<br/>(USART_TypeDef * USARTx, uint8_t Value)</code></b>                                         |
| Function description                              | Write in Transmitter Data Register (Transmit Data value, 8 bits)                                                                                   |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0xFF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TDR TDR LL_USART_TransmitData8</li> </ul>                                                                 |

### **LL\_USART\_TransmitData9**

|                                                   |                                                                                                                                                     |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_TransmitData9<br/>(USART_TypeDef * USARTx, uint16_t Value)</code></b>                                         |
| Function description                              | Write in Transmitter Data Register (Transmit Data value, 9 bits)                                                                                    |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>Value:</b> between Min_Data=0x00 and Max_Data=0x1FF</li> </ul> |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• TDR TDR LL_USART_TransmitData9</li> </ul>                                                                  |

### **LL\_USART\_RequestAutoBaudRate**

|                                                   |                                                                                                                                                                                                                                       |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_RequestAutoBaudRate<br/>(USART_TypeDef * USARTx)</code></b>                                                                                                                                     |
| Function description                              | Request an Automatic Baud Rate measurement on next received data frame.                                                                                                                                                               |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                     |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                       |
| Notes                                             | <ul style="list-style-type: none"> <li>• Macro<br/><code>IS_USART_AUTOBAUDRATE_DETECTION_INSTANCE(USARTx)</code> can be used to check whether or not Auto Baud Rate detection feature is supported by the USARTx instance.</li> </ul> |
| Reference Manual<br>to LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• RQR ABRRQ LL_USART_RequestAutoBaudRate</li> </ul>                                                                                                                                            |

**LL\_USART\_RequestBreakSending**

|                                                   |                                                                                                   |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_RequestBreakSending(<br/>USART_TypeDef * USARTx)</code></b> |
| Function description                              | Request Break sending.                                                                            |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>USARTx:</b> USART Instance</li></ul>                   |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                     |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• RQR SBKRQ LL_USART_RequestBreakSending</li></ul>          |

**LL\_USART\_RequestEnterMuteMode**

|                                                   |                                                                                                    |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_RequestEnterMuteMode(<br/>USART_TypeDef * USARTx)</code></b> |
| Function description                              | Put USART in mute mode and set the RWU flag.                                                       |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>USARTx:</b> USART Instance</li></ul>                    |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                      |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• RQR MMRQ LL_USART_RequestEnterMuteMode</li></ul>           |

**LL\_USART\_RequestRxDataFlush**

|                                                   |                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------|
| Function name                                     | <b><code>_STATIC_INLINE void LL_USART_RequestRxDataFlush(<br/>USART_TypeDef * USARTx)</code></b> |
| Function description                              | Request a Receive Data flush.                                                                    |
| Parameters                                        | <ul style="list-style-type: none"><li>• <b>USARTx:</b> USART Instance</li></ul>                  |
| Return values                                     | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                    |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"><li>• RQR RXFRQ LL_USART_RequestRxDataFlush</li></ul>          |

**LL\_USART\_RequestTxDataFlush**

|                      |                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>_STATIC_INLINE void LL_USART_RequestTxDataFlush(<br/>USART_TypeDef * USARTx)</code></b>                                                                                 |
| Function description | Request a Transmit data flush.                                                                                                                                                   |
| Parameters           | <ul style="list-style-type: none"><li>• <b>USARTx:</b> USART Instance</li></ul>                                                                                                  |
| Return values        | <ul style="list-style-type: none"><li>• <b>None</b></li></ul>                                                                                                                    |
| Notes                | <ul style="list-style-type: none"><li>• Macro IS_SMARTCARD_INSTANCE(USARTx) can be used to check whether or not Smartcard feature is supported by the USARTx instance.</li></ul> |

Reference Manual to  
LL API cross  
reference:

- RQR TXFRQ LL\_USART\_RequestTxDataFlush

### **LL\_USART\_DeInit**

|                      |                                                                                                                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_USART_DeInit (USART_TypeDef * USARTx)</b>                                                                                                                                                                                                         |
| Function description | De-initialize USART registers (Registers restored to their default values).                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> </ul>                                                                                                                                                                                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:             <ul style="list-style-type: none"> <li>– SUCCESS: USART registers are de-initialized</li> <li>– ERROR: USART registers are not de-initialized</li> </ul> </li> </ul> |

### **LL\_USART\_Init**

|                      |                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_USART_Init (USART_TypeDef * USARTx,<br/>LL_USART_InitTypeDef * USART_InitStruct)</b>                                                                                                                                                                                                                                                                                        |
| Function description | Initialize USART registers according to the specified parameters in USART_InitStruct.                                                                                                                                                                                                                                                                                                         |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>USART_InitStruct:</b> pointer to a LL_USART_InitTypeDef structure that contains the configuration information for the specified USART peripheral.</li> </ul>                                                                                                                                             |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value:             <ul style="list-style-type: none"> <li>– SUCCESS: USART registers are initialized according to USART_InitStruct content</li> <li>– ERROR: Problem occurred during USART Registers initialization</li> </ul> </li> </ul>                                                                        |
| Notes                | <ul style="list-style-type: none"> <li>• As some bits in USART configuration registers can only be written when the USART is disabled (USART_CR1_UE bit =0), USART IP should be in disabled state prior calling this function. Otherwise, ERROR result will be returned.</li> <li>• Baud rate value stored in USART_InitStruct BaudRate field, should be valid (different from 0).</li> </ul> |

### **LL\_USART\_StructInit**

|                      |                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_USART_StructInit (LL_USART_InitTypeDef *<br/>USART_InitStruct)</b>                                                                                        |
| Function description | Set each LL_USART_InitTypeDef field to default value.                                                                                                                |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USART_InitStruct:</b> pointer to a LL_USART_InitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                      |

**LL\_USART\_ClockInit**

|                      |                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_USART_ClockInit (USART_TypeDef * USARTx,<br/>LL_USART_ClockInitTypeDef * USART_ClockInitStruct)</b>                                                                                                                                                                                                                     |
| Function description | Initialize USART Clock related settings according to the specified parameters in the USART_ClockInitStruct.                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USARTx:</b> USART Instance</li> <li>• <b>USART_ClockInitStruct:</b> pointer to a LL_USART_ClockInitTypeDef structure that contains the Clock configuration information for the specified USART peripheral.</li> </ul>                                                                         |
| Return values        | <ul style="list-style-type: none"> <li>• <b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>– SUCCESS: USART registers related to Clock settings are initialized according to USART_ClockInitStruct content</li> <li>– ERROR: Problem occurred during USART Registers initialization</li> </ul> </li> </ul> |
| Notes                | <ul style="list-style-type: none"> <li>• As some bits in USART configuration registers can only be written when the USART is disabled (USART_CR1_UE bit =0), USART IP should be in disabled state prior calling this function. Otherwise, ERROR result will be returned.</li> </ul>                                                       |

**LL\_USART\_ClockStructInit**

|                      |                                                                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_USART_ClockStructInit<br/>(LL_USART_ClockInitTypeDef * USART_ClockInitStruct)</b>                                                                                   |
| Function description | Set each field of a LL_USART_ClockInitTypeDef type structure to default value.                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>USART_ClockInitStruct:</b> pointer to a LL_USART_ClockInitTypeDef structure whose fields will be set to default values.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                |

## 79.3 USART Firmware driver defines

### 79.3.1 USART

***Address Length Detection***

LL\_USART\_ADDRESS\_DETECT\_4B 4-bit address detection method selected

LL\_USART\_ADDRESS\_DETECT\_7B 7-bit address detection (in 8-bit data mode) method selected

***Autobaud Detection***

LL\_USART\_AUTOBAUD\_DETECT\_ON\_STARTBIT Measurement of the start bit is used to detect the baud rate

LL\_USART\_AUTOBAUD\_DETECT\_ON\_FALLINGEDGE Falling edge to falling edge measurement. Received frame must start with a single bit = 1 -&gt; Frame = Start1xxxxxxxx

|                                      |                                                                                                                               |
|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| LL_USART_AUTOBAUD_DETECT_ON_7F_FRAME | 0x7F frame detection                                                                                                          |
| LL_USART_AUTOBAUD_DETECT_ON_55_FRAME | 0x55 frame detection                                                                                                          |
| <b>Binary Data Inversion</b>         |                                                                                                                               |
| LL_USART_BINARY_LOGIC_POSITIVE       | Logical data from the data register are send/received in positive/direct logic. (1=H, 0=L)                                    |
| LL_USART_BINARY_LOGIC_NEGATIVE       | Logical data from the data register are send/received in negative/inverse logic. (1=L, 0=H). The parity bit is also inverted. |
| <b>Bit Order</b>                     |                                                                                                                               |
| LL_USART_BITORDER_LSBFIRST           | data is transmitted/received with data bit 0 first, following the start bit                                                   |
| LL_USART_BITORDER_MSBFIRST           | data is transmitted/received with the MSB first, following the start bit                                                      |
| <b>Clear Flags Defines</b>           |                                                                                                                               |
| LL_USART_ICR_PECF                    | Parity error flag                                                                                                             |
| LL_USART_ICR_FEKF                    | Framing error flag                                                                                                            |
| LL_USART_ICR_NCF                     | Noise detected flag                                                                                                           |
| LL_USART_ICR_ORECF                   | Overrun error flag                                                                                                            |
| LL_USART_ICR_IDLECF                  | Idle line detected flag                                                                                                       |
| LL_USART_ICR_TCCF                    | Transmission complete flag                                                                                                    |
| LL_USART_ICR_LBDCF                   | LIN break detection flag                                                                                                      |
| LL_USART_ICR_CTSCF                   | CTS flag                                                                                                                      |
| LL_USART_ICR_RTOCF                   | Receiver timeout flag                                                                                                         |
| LL_USART_ICR_EOBCF                   | End of block flag                                                                                                             |
| LL_USART_ICR_CMCF                    | Character match flag                                                                                                          |
| LL_USART_ICR_WUCF                    | Wakeup from Stop mode flag                                                                                                    |
| <b>Clock Signal</b>                  |                                                                                                                               |
| LL_USART_CLOCK_DISABLE               | Clock signal not provided                                                                                                     |
| LL_USART_CLOCK_ENABLE                | Clock signal provided                                                                                                         |
| <b>Datawidth</b>                     |                                                                                                                               |
| LL_USART_DATAWIDTH_8B                | 8 bits word length : Start bit, 8 data bits, n stop bits                                                                      |
| LL_USART_DATAWIDTH_9B                | 9 bits word length : Start bit, 9 data bits, n stop bits                                                                      |
| <b>Driver Enable Polarity</b>        |                                                                                                                               |
| LL_USART_DE_POLARITY_HIGH            | DE signal is active high                                                                                                      |
| LL_USART_DE_POLARITY_LOW             | DE signal is active low                                                                                                       |
| <b>Communication Direction</b>       |                                                                                                                               |
| LL_USART_DIRECTION_NONE              | Transmitter and Receiver are disabled                                                                                         |

|                          |                                                 |
|--------------------------|-------------------------------------------------|
| LL_USART_DIRECTION_RX    | Transmitter is disabled and Receiver is enabled |
| LL_USART_DIRECTION_TX    | Transmitter is enabled and Receiver is disabled |
| LL_USART_DIRECTION_TX_RX | Transmitter and Receiver are enabled            |

**DMA Register Data**

|                                |                                                    |
|--------------------------------|----------------------------------------------------|
| LL_USART_DMA_REG_DATA_TRANSMIT | Get address of data register used for transmission |
| LL_USART_DMA_REG_DATA_RECEIVE  | Get address of data register used for reception    |

**Get Flags Defines**

|                    |                                     |
|--------------------|-------------------------------------|
| LL_USART_ISR_PE    | Parity error flag                   |
| LL_USART_ISR_FE    | Framing error flag                  |
| LL_USART_ISR_NE    | Noise detected flag                 |
| LL_USART_ISR_ORE   | Overrun error flag                  |
| LL_USART_ISR_IDLE  | Idle line detected flag             |
| LL_USART_ISR_RXNE  | Read data register not empty flag   |
| LL_USART_ISR_TC    | Transmission complete flag          |
| LL_USART_ISR_TXE   | Transmit data register empty flag   |
| LL_USART_ISR_LBDF  | LIN break detection flag            |
| LL_USART_ISR_CTSIF | CTS interrupt flag                  |
| LL_USART_ISR_CTS   | CTS flag                            |
| LL_USART_ISR_RTOF  | Receiver timeout flag               |
| LL_USART_ISR_EOBF  | End of block flag                   |
| LL_USART_ISR_ABRE  | Auto baud rate error flag           |
| LL_USART_ISR_ABRF  | Auto baud rate flag                 |
| LL_USART_ISR_BUSY  | Busy flag                           |
| LL_USART_ISR_CMF   | Character match flag                |
| LL_USART_ISR_SBKF  | Send break flag                     |
| LL_USART_ISR_RWU   | Receiver wakeup from Mute mode flag |
| LL_USART_ISR_WUF   | Wakeup from Stop mode flag          |
| LL_USART_ISR_TEACK | Transmit enable acknowledge flag    |
| LL_USART_ISR_REACK | Receive enable acknowledge flag     |

**Hardware Control**

|                            |                                                                                        |
|----------------------------|----------------------------------------------------------------------------------------|
| LL_USART_HWCONTROL_NONE    | CTS and RTS hardware flow control disabled                                             |
| LL_USART_HWCONTROL_RTS     | RTS output enabled, data is only requested when there is space in the receive buffer   |
| LL_USART_HWCONTROL_CTS     | CTS mode enabled, data is only transmitted when the nCTS input is asserted (tied to 0) |
| LL_USART_HWCONTROL_RTS_CTS | CTS and RTS hardware flow control enabled                                              |

***IrDA Power***

|                            |                        |
|----------------------------|------------------------|
| LL_USART_IRDA_POWER_NORMAL | IrDA normal power mode |
| LL_USART_IRDA_POWER_LOW    | IrDA low power mode    |

***IT Defines***

|                     |                                               |
|---------------------|-----------------------------------------------|
| LL_USART_CR1_IDLEIE | IDLE interrupt enable                         |
| LL_USART_CR1_RXNEIE | Read data register not empty interrupt enable |
| LL_USART_CR1_TCIE   | Transmission complete interrupt enable        |
| LL_USART_CR1_TXEIE  | Transmit data register empty interrupt enable |
| LL_USART_CR1_PEIE   | Parity error                                  |
| LL_USART_CR1_CMIE   | Character match interrupt enable              |
| LL_USART_CR1_RTOIE  | Receiver timeout interrupt enable             |
| LL_USART_CR1_EOBIE  | End of Block interrupt enable                 |
| LL_USART_CR2_LBDIE  | LIN break detection interrupt enable          |
| LL_USART_CR3_EIE    | Error interrupt enable                        |
| LL_USART_CR3_CTSIE  | CTS interrupt enable                          |
| LL_USART_CR3_WUFIE  | Wakeup from Stop mode interrupt enable        |

***Last Clock Pulse***

|                                 |                                                                    |
|---------------------------------|--------------------------------------------------------------------|
| LL_USART_LASTCLKPULSE_NO_OUTPUT | The clock pulse of the last data bit is not output to the SCLK pin |
| LL_USART_LASTCLKPULSE_OUTPUT    | The clock pulse of the last data bit is output to the SCLK pin     |

***LIN Break Detection Length***

|                              |                                        |
|------------------------------|----------------------------------------|
| LL_USART_LINBREAK_DETECT_10B | 10-bit break detection method selected |
| LL_USART_LINBREAK_DETECT_11B | 11-bit break detection method selected |

***Oversampling***

|                          |                    |
|--------------------------|--------------------|
| LL_USART_OVERSAMPLING_16 | Oversampling by 16 |
| LL_USART_OVERSAMPLING_8  | Oversampling by 8  |

***Parity Control***

|                      |                                                    |
|----------------------|----------------------------------------------------|
| LL_USART_PARITY_NONE | Parity control disabled                            |
| LL_USART_PARITY EVEN | Parity control enabled and Even Parity is selected |
| LL_USART_PARITY ODD  | Parity control enabled and Odd Parity is selected  |

***Clock Phase***

|                      |                                                            |
|----------------------|------------------------------------------------------------|
| LL_USART_PHASE_1EDGE | The first clock transition is the first data capture edge  |
| LL_USART_PHASE_2EDGE | The second clock transition is the first data capture edge |

***Clock Polarity***

|                       |                                                          |
|-----------------------|----------------------------------------------------------|
| LL_USART_POLARITY_LOW | Steady low value on SCLK pin outside transmission window |
|-----------------------|----------------------------------------------------------|

**LL\_USART\_POLARITY\_HIGH** Steady high value on SCLK pin outside transmission window

#### **RX Pin Active Level Inversion**

**LL\_USART\_RXPIN\_LEVEL\_STANDARD** RX pin signal works using the standard logic levels

**LL\_USART\_RXPIN\_LEVEL\_INVERTED** RX pin signal values are inverted.

#### **Stop Bits**

**LL\_USART\_STOPBITS\_0\_5** 0.5 stop bit

**LL\_USART\_STOPBITS\_1** 1 stop bit

**LL\_USART\_STOPBITS\_1\_5** 1.5 stop bits

**LL\_USART\_STOPBITS\_2** 2 stop bits

#### **TX Pin Active Level Inversion**

**LL\_USART\_TXPIN\_LEVEL\_STANDARD** TX pin signal works using the standard logic levels

**LL\_USART\_TXPIN\_LEVEL\_INVERTED** TX pin signal values are inverted.

#### **TX RX Pins Swap**

**LL\_USART\_TXRX\_STANDARD** TX/RX pins are used as defined in standard pinout

**LL\_USART\_TXRX\_SWAPPED** TX and RX pins functions are swapped.

#### **Wakeup**

**LL\_USART\_WAKEUP\_IDLELINE** USART wake up from Mute mode on Idle Line

**LL\_USART\_WAKEUP\_ADDRESSMARK** USART wake up from Mute mode on Address Mark

#### **Wakeup Activation**

**LL\_USART\_WAKEUP\_ON\_ADDRESS** Wake up active on address match

**LL\_USART\_WAKEUP\_ON\_STARTBIT** Wake up active on Start bit detection

**LL\_USART\_WAKEUP\_ON\_RXNE** Wake up active on RXNE

#### **Exported Macros Helper**

**\_LL\_USART\_DIV\_SAMPLING8** **Description:**

- Compute USARTDIV value according to Peripheral Clock and expected Baud Rate in 8 bits sampling mode (32 bits value of USARTDIV is returned)

#### **Parameters:**

- **\_PERIPHCLK\_**: Peripheral Clock frequency used for USART instance
- **\_BAUDRATE\_**: Baud rate value to achieve

#### **Return value:**

- USARTDIV: value to be used for BRR register filling in OverSampling\_8 case

**\_LL\_USART\_DIV\_SAMPLING16 Description:**

- Compute USARTDIV value according to Peripheral Clock and expected Baud Rate in 16 bits sampling mode (32 bits value of USARTDIV is returned)

**Parameters:**

- PERIPHCLK: Peripheral Clock frequency used for USART instance
- BAUDRATE: Baud rate value to achieve

**Return value:**

- USARTDIV: value to be used for BRR register filling in OverSampling\_16 case

**Common Write and read registers Macros****LL\_USART\_WriteReg Description:**

- Write a value in USART register.

**Parameters:**

- INSTANCE: USART Instance
- REG: Register to be written
- VALUE: Value to be written in the register

**Return value:**

- None

**LL\_USART\_ReadReg Description:**

- Read a value in USART register.

**Parameters:**

- INSTANCE: USART Instance
- REG: Register to be read

**Return value:**

- Register: value

## 80 LL UTILS Generic Driver

### 80.1 UTILS Firmware driver registers structures

#### 80.1.1 LL\_UTILS\_PLLInitTypeDef

##### Data Fields

- *uint32\_t PLLMul*
- *uint32\_t Prediv*

##### Field Documentation

- *uint32\_t LL\_UTILS\_PLLInitTypeDef::PLLMul*  
Multiplication factor for PLL VCO input clock. This parameter can be a value of [RCC\\_LL\\_EC\\_PLL\\_MUL](#)This feature can be modified afterwards using unitary function [LL\\_RCC\\_PLL\\_ConfigDomain\\_SYS\(\)](#).
- *uint32\_t LL\_UTILS\_PLLInitTypeDef::Prediv*  
Division factor for HSE used as PLL clock source. This parameter can be a value of [RCC\\_LL\\_EC\\_PREDIV\\_DIV](#)This feature can be modified afterwards using unitary function [LL\\_RCC\\_PLL\\_ConfigDomain\\_SYS\(\)](#).

#### 80.1.2 LL\_UTILS\_ClkInitTypeDef

##### Data Fields

- *uint32\_t AHBCLKDivider*
- *uint32\_t APB1CLKDivider*
- *uint32\_t APB2CLKDivider*

##### Field Documentation

- *uint32\_t LL\_UTILS\_ClkInitTypeDef::AHBCLKDivider*  
The AHB clock (HCLK) divider. This clock is derived from the system clock (SYSCLK). This parameter can be a value of [RCC\\_LL\\_EC\\_SYSCLK\\_DIV](#)This feature can be modified afterwards using unitary function [LL\\_RCC\\_SetAHBPrescaler\(\)](#).
- *uint32\_t LL\_UTILS\_ClkInitTypeDef::APB1CLKDivider*  
The APB1 clock (PCLK1) divider. This clock is derived from the AHB clock (HCLK). This parameter can be a value of [RCC\\_LL\\_EC\\_APB1\\_DIV](#)This feature can be modified afterwards using unitary function [LL\\_RCC\\_SetAPB1Prescaler\(\)](#).
- *uint32\_t LL\_UTILS\_ClkInitTypeDef::APB2CLKDivider*  
The APB2 clock (PCLK2) divider. This clock is derived from the AHB clock (HCLK). This parameter can be a value of [RCC\\_LL\\_EC\\_APB2\\_DIV](#)This feature can be modified afterwards using unitary function [LL\\_RCC\\_SetAPB2Prescaler\(\)](#).

### 80.2 UTILS Firmware driver API description

#### 80.2.1 System Configuration functions

System, AHB and APB buses clocks configuration

- The maximum frequency of the SYSCLK, HCLK, PCLK1 and PCLK2 is 72000000 Hz.
- This section contains the following APIs:
- [LL\\_SetSystemCoreClock\(\)](#)
  - [LL\\_PLL\\_ConfigSystemClock\\_HSI\(\)](#)

- [\*\*LL\\_PLL\\_ConfigSystemClock\\_HSE\(\)\*\*](#)

## 80.2.2 Detailed description of functions

### **LL\_GetUID\_Word0**

|                      |                                                                                                                                |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_GetUID_Word0 (void )</code></b>                                                           |
| Function description | Get Word0 of the unique device identifier (UID based on 96 bits)                                                               |
| Return values        | <ul style="list-style-type: none"> <li>• <b>UID[31:0]:</b> X and Y coordinates on the wafer expressed in BCD format</li> </ul> |

### **LL\_GetUID\_Word1**

|                      |                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_GetUID_Word1 (void )</code></b>                                                              |
| Function description | Get Word1 of the unique device identifier (UID based on 96 bits)                                                                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>UID[63:32]:</b> Wafer number (UID[39:32]) &amp; LOT_NUM[23:0] (UID[63:40])</li> </ul> |

### **LL\_GetUID\_Word2**

|                      |                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_GetUID_Word2 (void )</code></b>                                               |
| Function description | Get Word2 of the unique device identifier (UID based on 96 bits)                                                   |
| Return values        | <ul style="list-style-type: none"> <li>• <b>UID[95:64]:</b> Lot number (ASCII encoded) - LOT_NUM[55:24]</li> </ul> |

### **LL\_GetFlashSize**

|                      |                                                                                                                                                                                     |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE uint32_t LL_GetFlashSize (void )</code></b>                                                                                                                |
| Function description | Get Flash memory size.                                                                                                                                                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>FLASH_SIZE[15:0]:</b> Flash memory size</li> </ul>                                                                                      |
| Notes                | <ul style="list-style-type: none"> <li>• This bitfield indicates the size of the device Flash memory expressed in Kbytes. As an example, 0x040 corresponds to 64 Kbytes.</li> </ul> |

### **LL\_InitTick**

|                      |                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b><code>__STATIC_INLINE void LL_InitTick (uint32_t HCLKFrequency, uint32_t Ticks)</code></b>                                                                                                                 |
| Function description | This function configures the Cortex-M SysTick source of the time base.                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HCLKFrequency:</b> HCLK frequency in Hz (can be calculated thanks to RCC helper macro)</li> <li>• <b>Ticks:</b> Number of ticks</li> </ul>                        |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>• When a RTOS is used, it is recommended to avoid changing the SysTick configuration by calling this function, for a delay use rather osDelay RTOS service.</li> </ul> |

**LL\_Init1msTick**

|                      |                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_Init1msTick (uint32_t HCLKFrequency)</b>                                                                                                                                                                                                                                                                         |
| Function description | This function configures the Cortex-M SysTick source to have 1ms time base.                                                                                                                                                                                                                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HCLKFrequency:</b> HCLK frequency in Hz</li> </ul>                                                                                                                                                                                                                              |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                             |
| Notes                | <ul style="list-style-type: none"> <li>• When a RTOS is used, it is recommended to avoid changing the Systick configuration by calling this function, for a delay use rather osDelay RTOS service.</li> <li>• HCLK frequency can be calculated thanks to RCC helper macro or function LL_RCC_GetSystemClocksFreq</li> </ul> |

**LL\_mDdelay**

|                      |                                                                                                                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_mDdelay (uint32_t Delay)</b>                                                                                                                                                                                                                                   |
| Function description | This function provides accurate delay (in milliseconds) based on SysTick counter flag.                                                                                                                                                                                    |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>Delay:</b> specifies the delay time length, in milliseconds.</li> </ul>                                                                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                           |
| Notes                | <ul style="list-style-type: none"> <li>• When a RTOS is used, it is recommended to avoid using blocking delay and use rather osDelay service.</li> <li>• To respect 1ms timebase, user should call LL_Init1msTick function which will configure Systick to 1ms</li> </ul> |

**LL\_SetSystemCoreClock**

|                      |                                                                                                                                               |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>void LL_SetSystemCoreClock (uint32_t HCLKFrequency)</b>                                                                                    |
| Function description | This function sets directly SystemCoreClock CMSIS variable.                                                                                   |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>HCLKFrequency:</b> HCLK frequency in Hz (can be calculated thanks to RCC helper macro)</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>• Variable can be calculated also through SystemCoreClockUpdate function.</li> </ul>                   |

**LL\_PLL\_ConfigSystemClock\_HSI**

|                      |                                                                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_PLL_ConfigSystemClock_HSI<br/>(LL_UTILS_PLLInitTypeDef * UTILS_PLLInitStruct,<br/>LL_UTILS_ClkInitTypeDef * UTILS_ClkInitStruct)</b>                                                                                                                                                                            |
| Function description | This function configures system clock with HSI as clock source of the PLL.                                                                                                                                                                                                                                                        |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>UTILS_PLLInitStruct:</b> pointer to a LL_UTILS_PLLInitTypeDef structure that contains the configuration information for the PLL.</li> <li>• <b>UTILS_ClkInitStruct:</b> pointer to a LL_UTILS_ClkInitTypeDef structure that contains the configuration information for the</li> </ul> |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | BUS prescalers.                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Return values | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>SUCCESS: Max frequency configuration done</li> <li>ERROR: Max frequency configuration not done</li> </ul> </li> </ul>                                                                                                                                                                                       |
| Notes         | <ul style="list-style-type: none"> <li>The application need to ensure that PLL is disabled.</li> <li>Function is based on the following formula: PLL output frequency = ((HSI frequency / PREDIV) * PLLMUL)PREDIV: Set to 2 for few devices</li> <li>PLLMUL: The application software must set correctly the PLL multiplication factor to not exceed 72MHz</li> <li>FLASH latency can be modified through this function.</li> </ul> |

### LL\_PLL\_ConfigSystemClock\_HSE

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <b>ErrorStatus LL_PLL_ConfigSystemClock_HSE (uint32_t HSEFrequency, uint32_t HSEBypass, LL_UTILS_PLLInitTypeDef * UTILS_PLLInitStruct, LL_UTILS_ClkInitTypeDef * UTILS_ClkInitStruct)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Function description | This function configures system clock with HSE as clock source of the PLL.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters           | <ul style="list-style-type: none"> <li><b>HSEFrequency:</b> Value between Min_Data = 4000000 and Max_Data = 32000000</li> <li><b>HSEBypass:</b> This parameter can be one of the following values: <ul style="list-style-type: none"> <li>LL_UTILS_HSEBYPASS_ON</li> <li>LL_UTILS_HSEBYPASS_OFF</li> </ul> </li> <li><b>UTILS_PLLInitStruct:</b> pointer to a LL_UTILS_PLLInitTypeDef structure that contains the configuration information for the PLL.</li> <li><b>UTILS_ClkInitStruct:</b> pointer to a LL_UTILS_ClkInitTypeDef structure that contains the configuration information for the BUS prescalers.</li> </ul> |
| Return values        | <ul style="list-style-type: none"> <li><b>An:</b> ErrorStatus enumeration value: <ul style="list-style-type: none"> <li>SUCCESS: Max frequency configuration done</li> <li>ERROR: Max frequency configuration not done</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                                                                                               |
| Notes                | <ul style="list-style-type: none"> <li>The application need to ensure that PLL is disabled.</li> <li>Function is based on the following formula: PLL output frequency = ((HSI frequency / PREDIV) * PLLMUL)PREDIV: Set to 2 for few devices</li> <li>PLLMUL: The application software must set correctly the PLL multiplication factor to not exceed UTILS_PLL_OUTPUT_MAX</li> <li>FLASH latency can be modified through this function.</li> </ul>                                                                                                                                                                          |

## 80.3 UTILS Firmware driver defines

### 80.3.1 UTILS

#### *HSE Bypass activation*

`LL_UTILS_HSEBYPASS_OFF` HSE Bypass is not enabled

`LL_UTILS_HSEBYPASS_ON` HSE Bypass is enabled

## 81 LL WWDG Generic Driver

### 81.1 WWDG Firmware driver API description

#### 81.1.1 Detailed description of functions

##### **LL\_WWDG\_Enable**

|                                                   |                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_WWDG_Enable (WWDG_TypeDef * WWDGx)</code>                                                                                                                                                                                                                          |
| Function description                              | Enable Window Watchdog.                                                                                                                                                                                                                                                                          |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>WWDGx:</b> WWDG Instance</li> </ul>                                                                                                                                                                                                                  |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                  |
| Notes                                             | <ul style="list-style-type: none"> <li>• It is enabled by setting the WDGA bit in the WWDG_CR register, then it cannot be disabled again except by a reset. This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR WDGA LL_WWDG_Enable</li> </ul>                                                                                                                                                                                                                       |

##### **LL\_WWDG\_IsEnabled**

|                                                   |                                                                                    |
|---------------------------------------------------|------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_WWDG_IsEnabled (WWDG_TypeDef * WWDGx)</code>     |
| Function description                              | Checks if Window Watchdog is enabled.                                              |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>WWDGx:</b> WWDG Instance</li> </ul>    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CR WDGA LL_WWDG_IsEnabled</li> </ul>      |

##### **LL\_WWDG\_SetCounter**

|                      |                                                                                                                                                                                                                                                                                                                |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE void LL_WWDG_SetCounter (WWDG_TypeDef * WWDGx, uint32_t Counter)</code>                                                                                                                                                                                                                  |
| Function description | Set the Watchdog counter value to provided value (7-bits T[6:0])                                                                                                                                                                                                                                               |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>WWDGx:</b> WWDG Instance</li> <li>• <b>Counter:</b> 0..0x7F (7 bit counter value)</li> </ul>                                                                                                                                                                       |
| Return values        | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                |
| Notes                | <ul style="list-style-type: none"> <li>• When writing to the WWDG_CR register, always write 1 in the MSB b6 to avoid generating an immediate reset. This counter is decremented every (4096 x 2<sup>exp(WDGTB)</sup>) PCLK cycles. A reset is produced when it rolls over from 0x40 to 0x3F (bit T6</li> </ul> |

becomes cleared) Setting the counter lower then 0x40 causes an immediate reset (if WWDG enabled)

Reference Manual to  
LL API cross  
reference:

- CR T LL\_WWDG\_SetCounter

### **LL\_WWDG\_GetCounter**

Function name

**`_STATIC_INLINE uint32_t LL_WWDG_GetCounter  
(WWDG_TypeDef * WWDGx)`**

Function description

Return current Watchdog Counter Value (7 bits counter value)

Parameters

- **WWDGx:** WWDG Instance

Return values

- **7:** bit Watchdog Counter value

Reference Manual to  
LL API cross  
reference:

- CR T LL\_WWDG\_GetCounter

### **LL\_WWDG\_SetPrescaler**

Function name

**`_STATIC_INLINE void LL_WWDG_SetPrescaler  
(WWDG_TypeDef * WWDGx, uint32_t Prescaler)`**

Function description

Set the time base of the prescaler (WDGTB).

Parameters

- **WWDGx:** WWDG Instance
- **Prescaler:** This parameter can be one of the following values:
  - LL\_WWDG\_PRESCALER\_1
  - LL\_WWDG\_PRESCALER\_2
  - LL\_WWDG\_PRESCALER\_4
  - LL\_WWDG\_PRESCALER\_8

Return values

- **None**

Notes

- Prescaler is used to apply ratio on PCLK clock, so that Watchdog counter is decremented every (4096 x 2<sup>expWDGTB</sup>) PCLK cycles

Reference Manual to  
LL API cross  
reference:

- CFR WDGTB LL\_WWDG\_SetPrescaler

### **LL\_WWDG\_GetPrescaler**

Function name

**`_STATIC_INLINE uint32_t LL_WWDG_GetPrescaler  
(WWDG_TypeDef * WWDGx)`**

Function description

Return current Watchdog Prescaler Value.

Parameters

- **WWDGx:** WWDG Instance

Return values

- **Returned:** value can be one of the following values:
  - LL\_WWDG\_PRESCALER\_1
  - LL\_WWDG\_PRESCALER\_2
  - LL\_WWDG\_PRESCALER\_4

- Reference Manual to  
LL API cross  
reference:
- LL\_WWDG\_PRESCALER\_8
  - CFR WDGTB LL\_WWDG\_GetPrescaler

### LL\_WWDG\_SetWindow

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE void LL_WWDG_SetWindow(<br/>(WWDG_TypeDef * WWDGx, uint32_t Window)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Function description                              | Set the Watchdog Window value to be compared to the downcounter (7-bits W[6:0]).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>WWDGx:</b> WWDG Instance</li> <li>• <b>Window:</b> 0x00..0x7F (7 bit Window value)</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>None</b></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Notes                                             | <ul style="list-style-type: none"> <li>• This window value defines when write in the WWDG_CR register to program Watchdog counter is allowed. Watchdog counter value update must occur only when the counter value is lower than the Watchdog window register value. Otherwise, a MCU reset is generated if the 7-bit Watchdog counter value (in the control register) is refreshed before the downcounter has reached the watchdog window register value. Physically is possible to set the Window lower than 0x40 but it is not recommended. To generate an immediate reset, it is possible to set the Counter lower than 0x40.</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFR W LL_WWDG_SetWindow</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

### LL\_WWDG\_GetWindow

|                                                   |                                                                                         |
|---------------------------------------------------|-----------------------------------------------------------------------------------------|
| Function name                                     | <code>__STATIC_INLINE uint32_t LL_WWDG_GetWindow(<br/>(WWDG_TypeDef * WWDGx)</code>     |
| Function description                              | Return current Watchdog Window Value (7 bits value)                                     |
| Parameters                                        | <ul style="list-style-type: none"> <li>• <b>WWDGx:</b> WWDG Instance</li> </ul>         |
| Return values                                     | <ul style="list-style-type: none"> <li>• <b>7:</b> bit Watchdog Window value</li> </ul> |
| Reference Manual to<br>LL API cross<br>reference: | <ul style="list-style-type: none"> <li>• CFR W LL_WWDG_SetWindow</li> </ul>             |

### LL\_WWDG\_IsActiveFlag\_EWKUP

|                      |                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------|
| Function name        | <code>__STATIC_INLINE uint32_t LL_WWDG_IsActiveFlag_EWKUP(<br/>(WWDG_TypeDef * WWDGx)</code>                     |
| Function description | Indicates if the WWDG Early Wakeup Interrupt Flag is set or not.                                                 |
| Parameters           | <ul style="list-style-type: none"> <li>• <b>WWDGx:</b> WWDG Instance</li> </ul>                                  |
| Return values        | <ul style="list-style-type: none"> <li>• <b>State:</b> of bit (1 or 0).</li> </ul>                               |
| Notes                | <ul style="list-style-type: none"> <li>• This bit is set by hardware when the counter has reached the</li> </ul> |

value 0x40. It must be cleared by software by writing 0. A write of 1 has no effect. This bit is also set if the interrupt is not enabled.

Reference Manual to  
LL API cross  
reference:

- SR EWIF LL\_WWDG\_IsActiveFlag\_EWKUP

### **LL\_WWDG\_ClearFlag\_EWKUP**

Function name **`__STATIC_INLINE void LL_WWDG_ClearFlag_EWKUP(  
(WWDG_TypeDef * WWDGx)`**

Function description Clear WWDG Early Wakeup Interrupt Flag (EWIF)

Parameters • **WWDGx:** WWDG Instance

Return values • **None**

Reference Manual to  
LL API cross  
reference:

- SR EWIF LL\_WWDG\_ClearFlag\_EWKUP

### **LL\_WWDG\_EnableIT\_EWKUP**

Function name **`__STATIC_INLINE void LL_WWDG_EnableIT_EWKUP(  
(WWDG_TypeDef * WWDGx)`**

Function description Enable the Early Wakeup Interrupt.

Parameters • **WWDGx:** WWDG Instance

Return values • **None**

Notes • When set, an interrupt occurs whenever the counter reaches value 0x40. This interrupt is only cleared by hardware after a reset

Reference Manual to  
LL API cross  
reference:

- CFR EWI LL\_WWDG\_EnableIT\_EWKUP

### **LL\_WWDG\_IsEnabledIT\_EWKUP**

Function name **`__STATIC_INLINE uint32_t LL_WWDG_IsEnabledIT_EWKUP(  
(WWDG_TypeDef * WWDGx)`**

Function description Check if Early Wakeup Interrupt is enabled.

Parameters • **WWDGx:** WWDG Instance

Return values • **State:** of bit (1 or 0).

Reference Manual to  
LL API cross  
reference:

- CFR EWI LL\_WWDG\_IsEnabledIT\_EWKUP

## 81.2 WWDG Firmware driver defines

### 81.2.1 WWDG

#### *IT Defines*

`LL_WWDG_CFR_EWI`

#### *PRESCALER*

`LL_WWDG_PRESCALER_1` WWDG counter clock = (PCLK1/4096)/1

`LL_WWDG_PRESCALER_2` WWDG counter clock = (PCLK1/4096)/2

`LL_WWDG_PRESCALER_4` WWDG counter clock = (PCLK1/4096)/4

`LL_WWDG_PRESCALER_8` WWDG counter clock = (PCLK1/4096)/8

#### *Common Write and read registers macros*

`LL_WWDG_WriteReg` **Description:**

- Write a value in WWDG register.

#### **Parameters:**

- `_INSTANCE_`: WWDG Instance
- `_REG_`: Register to be written
- `_VALUE_`: Value to be written in the register

#### **Return value:**

- None

`LL_WWDG_ReadReg` **Description:**

- Read a value in WWDG register.

#### **Parameters:**

- `_INSTANCE_`: WWDG Instance
- `_REG_`: Register to be read

#### **Return value:**

- Register: value

## 82 Correspondence between API registers and API low-layer driver functions

### 82.1 ADC

Table 25: Correspondence between ADC registers and ADC low-layer driver functions

| Register | Field     | Function                                    |
|----------|-----------|---------------------------------------------|
| AWD2CR   | AWD2CH    | <i>LL_ADC_GetAnalogWDMonitChannels</i>      |
|          |           | <i>LL_ADC_SetAnalogWDMonitChannels</i>      |
| AWD3CR   | AWD3CH    | <i>LL_ADC_GetAnalogWDMonitChannels</i>      |
|          |           | <i>LL_ADC_SetAnalogWDMonitChannels</i>      |
| CALFACT  | CALFACT_D | <i>LL_ADC_GetCalibrationFactor</i>          |
|          |           | <i>LL_ADC_SetCalibrationFactor</i>          |
|          | CALFACT_S | <i>LL_ADC_GetCalibrationFactor</i>          |
|          |           | <i>LL_ADC_SetCalibrationFactor</i>          |
| CCR      | CKMODE    | <i>LL_ADC_GetCommonClock</i>                |
|          |           | <i>LL_ADC_SetCommonClock</i>                |
|          | DELAY     | <i>LL_ADC_GetMultiTwoSamplingDelay</i>      |
|          |           | <i>LL_ADC_SetMultiTwoSamplingDelay</i>      |
|          | DMACFG    | <i>LL_ADC_GetMultiDMATransfer</i>           |
|          |           | <i>LL_ADC_SetMultiDMATransfer</i>           |
|          | DUAL      | <i>LL_ADC_GetMultimode</i>                  |
|          |           | <i>LL_ADC_SetMultimode</i>                  |
|          | MDMA      | <i>LL_ADC_GetMultiDMATransfer</i>           |
|          |           | <i>LL_ADC_SetMultiDMATransfer</i>           |
|          | PRESC     | <i>LL_ADC_GetCommonClock</i>                |
|          |           | <i>LL_ADC_SetCommonClock</i>                |
|          | TSEN      | <i>LL_ADC_GetCommonPathInternalCh</i>       |
|          |           | <i>LL_ADC_SetCommonPathInternalCh</i>       |
|          | VBATEN    | <i>LL_ADC_GetCommonPathInternalCh</i>       |
|          |           | <i>LL_ADC_SetCommonPathInternalCh</i>       |
|          | VREFEN    | <i>LL_ADC_GetCommonPathInternalCh</i>       |
|          |           | <i>LL_ADC_SetCommonPathInternalCh</i>       |
| CDR      | RDATA_MST | <i>LL_ADC_DMA_GetRegAddr</i>                |
|          |           | <i>LL_ADC_REG_ReadMultiConversionData32</i> |
|          | RDATA_SLV | <i>LL_ADC_DMA_GetRegAddr</i>                |
|          |           | <i>LL_ADC_REG_ReadMultiConversionData32</i> |

| Register | Field   | Function                                 |
|----------|---------|------------------------------------------|
| CFGREG   | ALIGN   | <i>LL_ADC_GetDataAlignment</i>           |
|          |         | <i>LL_ADC_SetDataAlignment</i>           |
|          | AUTDLY  | <i>LL_ADC_GetLowPowerMode</i>            |
|          |         | <i>LL_ADC_SetLowPowerMode</i>            |
|          | AWD1CH  | <i>LL_ADC_GetAnalogWDMonitChannels</i>   |
|          |         | <i>LL_ADC_SetAnalogWDMonitChannels</i>   |
|          | AWD1EN  | <i>LL_ADC_GetAnalogWDMonitChannels</i>   |
|          |         | <i>LL_ADC_SetAnalogWDMonitChannels</i>   |
|          | AWD1SGL | <i>LL_ADC_GetAnalogWDMonitChannels</i>   |
|          |         | <i>LL_ADC_SetAnalogWDMonitChannels</i>   |
|          | CONT    | <i>LL_ADC_REG_GetContinuousMode</i>      |
|          |         | <i>LL_ADC_REG_SetContinuousMode</i>      |
|          | DISCEN  | <i>LL_ADC_REG_GetSequencerDiscont</i>    |
|          |         | <i>LL_ADC_REG_SetSequencerDiscont</i>    |
|          | DISCNUM | <i>LL_ADC_REG_GetSequencerDiscont</i>    |
|          |         | <i>LL_ADC_REG_SetSequencerDiscont</i>    |
|          | DMACFG  | <i>LL_ADC_REG_GetDMATransfer</i>         |
|          |         | <i>LL_ADC_REG_SetDMATransfer</i>         |
|          | DMAEN   | <i>LL_ADC_REG_GetDMATransfer</i>         |
|          |         | <i>LL_ADC_REG_SetDMATransfer</i>         |
|          | EXTEN   | <i>LL_ADC_REG_GetTriggerEdge</i>         |
|          |         | <i>LL_ADC_REG_SetTriggerEdge</i>         |
|          |         | <i>LL_ADC_REG_IsTriggerSourceSWStart</i> |
|          |         | <i>LL_ADC_REG_SetTriggerSource</i>       |
|          |         | <i>LL_ADC_REG_SetTriggerSource</i>       |
|          | EXTSEL  | <i>LL_ADC_REG_SetTriggerSource</i>       |
|          |         | <i>LL_ADC_REG_SetTriggerSource</i>       |
|          | JAUTO   | <i>LL_ADC_INJ_SetTrigAuto</i>            |
|          |         | <i>LL_ADC_INJ_SetTrigAuto</i>            |
|          | JAWD1EN | <i>LL_ADC_GetAnalogWDMonitChannels</i>   |
|          |         | <i>LL_ADC_SetAnalogWDMonitChannels</i>   |
|          | JDISCEN | <i>LL_ADC_INJ_SetSequencerDiscont</i>    |
|          |         | <i>LL_ADC_INJ_SetSequencerDiscont</i>    |
|          | JQM     | <i>LL_ADC_INJ_SetQueueMode</i>           |
|          |         | <i>LL_ADC_INJ_SetQueueMode</i>           |
|          | OVRMOD  | <i>LL_ADC_REG_GetOverrun</i>             |

| Register | Field     | Function                                  |
|----------|-----------|-------------------------------------------|
| CR       | RES       | <i>LL_ADC_REG_SetOverrun</i>              |
|          |           | <i>LL_ADC_GetResolution</i>               |
|          |           | <i>LL_ADC_SetResolution</i>               |
|          | ADCAL     | <i>LL_ADC_IsCalibrationOnGoing</i>        |
|          |           | <i>LL_ADC_StartCalibration</i>            |
|          | ADCALDIF  | <i>LL_ADC_StartCalibration</i>            |
|          | ADDIS     | <i>LL_ADC_Disable</i>                     |
|          |           | <i>LL_ADC_IsDisableOngoing</i>            |
|          | ADEN      | <i>LL_ADC_Enable</i>                      |
|          |           | <i>LL_ADC_IsEnabled</i>                   |
|          | ADSTART   | <i>LL_ADC_REG_IsConversionOngoing</i>     |
|          |           | <i>LL_ADC_REG_StartConversion</i>         |
|          | ADSTP     | <i>LL_ADC_REG_IsStopConversionOngoing</i> |
|          |           | <i>LL_ADC_REG_StopConversion</i>          |
| CSR      | ADVREGEN  | <i>LL_ADC_DisableInternalRegulator</i>    |
|          |           | <i>LL_ADC_EnableInternalRegulator</i>     |
|          |           | <i>LL_ADC_IsInternalRegulatorEnabled</i>  |
|          | JADSTART  | <i>LL_ADC_INJ_IsConversionOngoing</i>     |
|          |           | <i>LL_ADC_INJ_StartConversion</i>         |
|          | JADSTP    | <i>LL_ADC_INJ_IsStopConversionOngoing</i> |
|          |           | <i>LL_ADC_INJ_StopConversion</i>          |
|          | ADRDY_MST | <i>LL_ADC_IsActiveFlag_MST_ADRDY</i>      |
|          | ADRDY_SLV | <i>LL_ADC_IsActiveFlag_SLV_ADRDY</i>      |
|          | AWD1_MST  | <i>LL_ADC_IsActiveFlag_MST_AWD1</i>       |
|          | AWD1_SLV  | <i>LL_ADC_IsActiveFlag_SLV_AWD1</i>       |
|          | AWD2_MST  | <i>LL_ADC_IsActiveFlag_MST_AWD2</i>       |
|          | AWD2_SLV  | <i>LL_ADC_IsActiveFlag_SLV_AWD2</i>       |
|          | AWD3_MST  | <i>LL_ADC_IsActiveFlag_MST_AWD3</i>       |
|          | AWD3_SLV  | <i>LL_ADC_IsActiveFlag_SLV_AWD3</i>       |
|          | EOC_MST   | <i>LL_ADC_IsActiveFlag_MST_EOC</i>        |
|          | EOC_SLV   | <i>LL_ADC_IsActiveFlag_SLV_EOC</i>        |
|          | EOSMP_MST | <i>LL_ADC_IsActiveFlag_MST_EOSMP</i>      |
|          | EOSMP_SLV | <i>LL_ADC_IsActiveFlag_SLV_EOSMP</i>      |
|          | EOS_MST   | <i>LL_ADC_IsActiveFlag_MST_EOS</i>        |
|          | EOS_SLV   | <i>LL_ADC_IsActiveFlag_SLV_EOS</i>        |
|          | JEOC_MST  | <i>LL_ADC_IsActiveFlag_MST_JEOC</i>       |

| Register | Field     | Function                               |
|----------|-----------|----------------------------------------|
| DR       | JEOC_SLV  | <i>LL_ADC_IsActiveFlag_SLV_JEOC</i>    |
|          | JEOS_MST  | <i>LL_ADC_IsActiveFlag_MST_JEOS</i>    |
|          | JEOS_SLV  | <i>LL_ADC_IsActiveFlag_SLV_JEOS</i>    |
|          | JQOVF_MST | <i>LL_ADC_IsActiveFlag_MST_JQOVF</i>   |
|          | JQOVF_SLV | <i>LL_ADC_IsActiveFlag_SLV_JQOVF</i>   |
|          | OVR_MST   | <i>LL_ADC_IsActiveFlag_MST_OVR</i>     |
|          | OVR_SLV   | <i>LL_ADC_IsActiveFlag_SLV_OVR</i>     |
| DIFSEL   | DIFSEL    | <i>LL_ADC_GetChannelSamplingTime</i>   |
| IER      | RDATA     | <i>LL_ADC_DMA_GetRegAddr</i>           |
|          |           | <i>LL_ADC_REG_ReadConversionData10</i> |
|          |           | <i>LL_ADC_REG_ReadConversionData12</i> |
|          |           | <i>LL_ADC_REG_ReadConversionData32</i> |
|          |           | <i>LL_ADC_REG_ReadConversionData6</i>  |
|          |           | <i>LL_ADC_REG_ReadConversionData8</i>  |
| IER      | ADRDYIE   | <i>LL_ADC_DisableIT_ADRDY</i>          |
|          |           | <i>LL_ADC_EnableIT_ADRDY</i>           |
|          |           | <i>LL_ADC_IsEnabledIT_ADRDY</i>        |
|          | AWD1IE    | <i>LL_ADC_DisableIT_AWD1</i>           |
|          |           | <i>LL_ADC_EnableIT_AWD1</i>            |
|          |           | <i>LL_ADC_IsEnabledIT_AWD1</i>         |
|          | AWD2IE    | <i>LL_ADC_DisableIT_AWD2</i>           |
|          |           | <i>LL_ADC_EnableIT_AWD2</i>            |
|          |           | <i>LL_ADC_IsEnabledIT_AWD2</i>         |
|          | AWD3IE    | <i>LL_ADC_DisableIT_AWD3</i>           |
|          |           | <i>LL_ADC_EnableIT_AWD3</i>            |
|          |           | <i>LL_ADC_IsEnabledIT_AWD3</i>         |
|          | EOCIE     | <i>LL_ADC_DisableIT_EOC</i>            |
|          |           | <i>LL_ADC_EnableIT_EOC</i>             |
|          |           | <i>LL_ADC_IsEnabledIT_EOC</i>          |
|          | EOSIE     | <i>LL_ADC_DisableIT_EOS</i>            |
|          |           | <i>LL_ADC_EnableIT_EOS</i>             |
|          |           | <i>LL_ADC_IsEnabledIT_EOS</i>          |
|          | EOSMPIE   | <i>LL_ADC_DisableIT_EOSMP</i>          |
|          |           | <i>LL_ADC_EnableIT_EOSMP</i>           |
|          |           | <i>LL_ADC_IsEnabledIT_EOSMP</i>        |
|          | JEOCIE    | <i>LL_ADC_DisableIT_JEOC</i>           |

| Register | Field   | Function                               |
|----------|---------|----------------------------------------|
| ISR      | JEOSIE  | <i>LL_ADC_EnableIT_JEOC</i>            |
|          |         | <i>LL_ADC_IsEnabledIT_JEOC</i>         |
|          |         | <i>LL_ADC_DisableIT_JEOS</i>           |
|          |         | <i>LL_ADC_EnableIT_JEOS</i>            |
|          | JQOVFIE | <i>LL_ADC_IsEnabledIT_JEOS</i>         |
|          |         | <i>LL_ADC_DisableIT_JQOVF</i>          |
|          |         | <i>LL_ADC_EnableIT_JQOVF</i>           |
|          | OVRIE   | <i>LL_ADC_IsEnabledIT_JQOVF</i>        |
|          |         | <i>LL_ADC_DisableIT_OVR</i>            |
|          |         | <i>LL_ADC_EnableIT_OVR</i>             |
|          |         | <i>LL_ADC_IsEnabledIT_OVR</i>          |
|          | ADRDY   | <i>LL_ADC_ClearFlag_ADRDY</i>          |
|          |         | <i>LL_ADC_IsActiveFlag_ADRDY</i>       |
|          | AWD1    | <i>LL_ADC_ClearFlag_AWD1</i>           |
|          |         | <i>LL_ADC_IsActiveFlag_AWD1</i>        |
|          | AWD2    | <i>LL_ADC_ClearFlag_AWD2</i>           |
|          |         | <i>LL_ADC_IsActiveFlag_AWD2</i>        |
|          | AWD3    | <i>LL_ADC_ClearFlag_AWD3</i>           |
|          |         | <i>LL_ADC_IsActiveFlag_AWD3</i>        |
|          | EOC     | <i>LL_ADC_ClearFlag_EOC</i>            |
|          |         | <i>LL_ADC_IsActiveFlag_EOC</i>         |
|          | EOS     | <i>LL_ADC_ClearFlag_EOS</i>            |
|          |         | <i>LL_ADC_IsActiveFlag_EOS</i>         |
|          | EOSMP   | <i>LL_ADC_ClearFlag_EOSMP</i>          |
|          |         | <i>LL_ADC_IsActiveFlag_EOSMP</i>       |
|          | JEOC    | <i>LL_ADC_ClearFlag_JEOC</i>           |
|          |         | <i>LL_ADC_IsActiveFlag_JEOC</i>        |
|          | JEOS    | <i>LL_ADC_ClearFlag_JEOS</i>           |
|          |         | <i>LL_ADC_IsActiveFlag_JEOS</i>        |
|          | JQOVF   | <i>LL_ADC_ClearFlag_JQOVF</i>          |
|          |         | <i>LL_ADC_IsActiveFlag_JQOVF</i>       |
|          | OVR     | <i>LL_ADC_ClearFlag_OVR</i>            |
|          |         | <i>LL_ADC_IsActiveFlag_OVR</i>         |
| JDR1     | JDAT    | <i>LL_ADC_INJ_ReadConversionData10</i> |
|          |         | <i>LL_ADC_INJ_ReadConversionData12</i> |
|          |         | <i>LL_ADC_INJ_ReadConversionData32</i> |

| Register | Field   | Function                                 |
|----------|---------|------------------------------------------|
|          |         | <i>LL_ADC_INJ_ReadConversionData6</i>    |
|          |         | <i>LL_ADC_INJ_ReadConversionData8</i>    |
| JDR2     | JDATA   | <i>LL_ADC_INJ_ReadConversionData10</i>   |
|          |         | <i>LL_ADC_INJ_ReadConversionData12</i>   |
|          |         | <i>LL_ADC_INJ_ReadConversionData32</i>   |
|          |         | <i>LL_ADC_INJ_ReadConversionData6</i>    |
|          |         | <i>LL_ADC_INJ_ReadConversionData8</i>    |
| JDR3     | JDATA   | <i>LL_ADC_INJ_ReadConversionData10</i>   |
|          |         | <i>LL_ADC_INJ_ReadConversionData12</i>   |
|          |         | <i>LL_ADC_INJ_ReadConversionData32</i>   |
|          |         | <i>LL_ADC_INJ_ReadConversionData6</i>    |
|          |         | <i>LL_ADC_INJ_ReadConversionData8</i>    |
| JDR4     | JDATA   | <i>LL_ADC_INJ_ReadConversionData10</i>   |
|          |         | <i>LL_ADC_INJ_ReadConversionData12</i>   |
|          |         | <i>LL_ADC_INJ_ReadConversionData32</i>   |
|          |         | <i>LL_ADC_INJ_ReadConversionData6</i>    |
|          |         | <i>LL_ADC_INJ_ReadConversionData8</i>    |
| JSQR     | JEXTEN  | <i>LL_ADC_INJ_ConfigQueueContext</i>     |
|          |         | <i>LL_ADC_INJ_GetTriggerEdge</i>         |
|          |         | <i>LL_ADC_INJ_GetTriggerSource</i>       |
|          |         | <i>LL_ADC_INJ_IsTriggerSourceSWStart</i> |
|          |         | <i>LL_ADC_INJ_SetTriggerEdge</i>         |
|          |         | <i>LL_ADC_INJ_SetTriggerSource</i>       |
|          | JEXTSEL | <i>LL_ADC_INJ_ConfigQueueContext</i>     |
|          |         | <i>LL_ADC_INJ_GetTriggerSource</i>       |
|          |         | <i>LL_ADC_INJ_SetTriggerSource</i>       |
|          | JL      | <i>LL_ADC_INJ_ConfigQueueContext</i>     |
|          |         | <i>LL_ADC_INJ_GetSequencerLength</i>     |
|          |         | <i>LL_ADC_INJ_SetSequencerLength</i>     |
|          | JSQ1    | <i>LL_ADC_INJ_ConfigQueueContext</i>     |
|          |         | <i>LL_ADC_INJ_GetSequencerRanks</i>      |
|          |         | <i>LL_ADC_INJ_SetSequencerRanks</i>      |
|          | JSQ2    | <i>LL_ADC_INJ_ConfigQueueContext</i>     |
|          |         | <i>LL_ADC_INJ_GetSequencerRanks</i>      |
|          |         | <i>LL_ADC_INJ_SetSequencerRanks</i>      |
|          | JSQ3    | <i>LL_ADC_INJ_ConfigQueueContext</i>     |

| Register | Field      | Function                             |
|----------|------------|--------------------------------------|
| JSQ4     |            | <i>LL_ADC_INJ_GetSequencerRanks</i>  |
|          |            | <i>LL_ADC_INJ_SetSequencerRanks</i>  |
|          | JSQ4       | <i>LL_ADC_INJ_ConfigQueueContext</i> |
|          |            | <i>LL_ADC_INJ_GetSequencerRanks</i>  |
|          |            | <i>LL_ADC_INJ_SetSequencerRanks</i>  |
| OFR1     | OFFSET1    | <i>LL_ADC_GetOffsetLevel</i>         |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          | OFFSET1_CH | <i>LL_ADC_GetOffsetChannel</i>       |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          | OFFSET1_EN | <i>LL_ADC_GetOffsetState</i>         |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          |            | <i>LL_ADC_SetOffsetState</i>         |
| OFR2     | OFFSET2    | <i>LL_ADC_GetOffsetLevel</i>         |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          | OFFSET2_CH | <i>LL_ADC_GetOffsetChannel</i>       |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          | OFFSET2_EN | <i>LL_ADC_GetOffsetState</i>         |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          |            | <i>LL_ADC_SetOffsetState</i>         |
| OFR3     | OFFSET3    | <i>LL_ADC_GetOffsetLevel</i>         |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          | OFFSET3_CH | <i>LL_ADC_GetOffsetChannel</i>       |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          | OFFSET3_EN | <i>LL_ADC_GetOffsetState</i>         |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          |            | <i>LL_ADC_SetOffsetState</i>         |
| OFR4     | OFFSET4    | <i>LL_ADC_GetOffsetLevel</i>         |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          | OFFSET4_CH | <i>LL_ADC_GetOffsetChannel</i>       |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          | OFFSET4_EN | <i>LL_ADC_GetOffsetState</i>         |
|          |            | <i>LL_ADC_SetOffset</i>              |
|          |            | <i>LL_ADC_SetOffsetState</i>         |
| SMPR1    | SMP0       | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |            | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP1       | <i>LL_ADC_GetChannelSamplingTime</i> |

| Register | Field | Function                             |
|----------|-------|--------------------------------------|
| SMPR2    | SMP2  | <i>LL_ADC_SetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP3  | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP4  | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP5  | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP6  | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP7  | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP8  | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP9  | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
| SMPR2    | SMP10 | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP11 | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP12 | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP13 | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP14 | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP15 | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP16 | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP17 | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
|          | SMP18 | <i>LL_ADC_GetChannelSamplingTime</i> |
|          |       | <i>LL_ADC_SetChannelSamplingTime</i> |
| SQR1     | L     | <i>LL_ADC_REG_GetSequencerLength</i> |

| Register | Field | Function                               |
|----------|-------|----------------------------------------|
| SQR1     | SQ1   | <i>LL_ADC_REG_SetSequencerLength</i>   |
|          |       | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ2   | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ3   | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ4   | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
| SQR2     | SQ5   | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ6   | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ7   | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ8   | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ9   | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
| SQR3     | SQ10  | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ11  | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ12  | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ13  | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
| SQR4     | SQ14  | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
|          | SQ15  | <i>LL_ADC_REG_GetSequencerRanks</i>    |
|          |       | <i>LL_ADC_REG_SetSequencerRanks</i>    |
| TR1      | HT1   | <i>LL_ADC_ConfigAnalogWDThresholds</i> |
|          |       | <i>LL_ADC_GetAnalogWDThresholds</i>    |
|          |       | <i>LL_ADC_SetAnalogWDThresholds</i>    |

| Register | Field | Function                               |
|----------|-------|----------------------------------------|
|          | LT1   | <i>LL_ADC_ConfigAnalogWDThresholds</i> |
|          |       | <i>LL_ADC_GetAnalogWDThresholds</i>    |
|          |       | <i>LL_ADC_SetAnalogWDThresholds</i>    |
| TR2      | HT2   | <i>LL_ADC_ConfigAnalogWDThresholds</i> |
|          |       | <i>LL_ADC_GetAnalogWDThresholds</i>    |
|          |       | <i>LL_ADC_SetAnalogWDThresholds</i>    |
| TR3      | LT2   | <i>LL_ADC_ConfigAnalogWDThresholds</i> |
|          |       | <i>LL_ADC_GetAnalogWDThresholds</i>    |
|          |       | <i>LL_ADC_SetAnalogWDThresholds</i>    |
|          | HT3   | <i>LL_ADC_ConfigAnalogWDThresholds</i> |
|          |       | <i>LL_ADC_GetAnalogWDThresholds</i>    |
|          |       | <i>LL_ADC_SetAnalogWDThresholds</i>    |
|          | LT3   | <i>LL_ADC_ConfigAnalogWDThresholds</i> |
|          |       | <i>LL_ADC_GetAnalogWDThresholds</i>    |
|          |       | <i>LL_ADC_SetAnalogWDThresholds</i>    |

## 82.2 BUS

Table 26: Correspondence between BUS registers and BUS low-layer driver functions

| Register | Field   | Function                           |
|----------|---------|------------------------------------|
| AHBENR   | ADC12EN | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
|          | ADC1EN  | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
|          | ADC34EN | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
|          | CRCEN   | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
|          | DMA1EN  | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
|          | DMA2EN  | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_AHB1_GRP1_EnableClock</i>    |

| Register | Field | Function                           |
|----------|-------|------------------------------------|
| FLITFEN  |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
|          |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| FMCEN    |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| GPIOAEN  |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| GPIOBEN  |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| GPIOCEN  |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| GPIODEN  |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| GPIOEEN  |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| GPIOFEN  |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| GPIOGEN  |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| GPIOHEN  |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| SRAMEN   |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |
|          |       | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
| TSCEN    |       | <i>LL_AHB1_GRP1_DisableClock</i>   |
|          |       | <i>LL_AHB1_GRP1_EnableClock</i>    |

| Register | Field    | Function                           |
|----------|----------|------------------------------------|
| AHBRSTR  | ADC12RST | <i>LL_AHB1_GRP1_IsEnabledClock</i> |
|          |          | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | ADC1RST  | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | ADC34RST | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | FMCRST   | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | GPIOARST | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | GPIOBRST | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | GPIOCRST | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | GPIODRST | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | GPIOERST | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | GPIOFRST | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | GPIOGRST | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | GPIOHRST | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
|          | TSCRST   | <i>LL_AHB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_AHB1_GRP1_ReleaseReset</i>   |
| APB1ENR  | CANEN    | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | CECEN    | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | DAC1EN   | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |

| Register | Field   | Function                           |
|----------|---------|------------------------------------|
|          | DAC2EN  | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | I2C1EN  | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | I2C2EN  | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | I2C3EN  | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | PWREN   | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | SPI2EN  | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | SPI3EN  | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | TIM12EN | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | TIM13EN | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | TIM14EN | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | TIM18EN | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | TIM2EN  | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |         | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |         | <i>LL_APB1_GRP1_IsEnabledClock</i> |

| Register | Field    | Function                           |
|----------|----------|------------------------------------|
|          | TIM3EN   | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | TIM4EN   | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | TIM5EN   | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | TIM6EN   | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | TIM7EN   | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | UART4EN  | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | UART5EN  | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | USART2EN | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | USART3EN | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | USBEN    | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | WWDGGEN  | <i>LL_APB1_GRP1_DisableClock</i>   |
|          |          | <i>LL_APB1_GRP1_EnableClock</i>    |
|          |          | <i>LL_APB1_GRP1_IsEnabledClock</i> |
|          | CANRST   | <i>LL_APB1_GRP1_ForceReset</i>     |
|          |          | <i>LL_APB1_GRP1_ReleaseReset</i>   |
|          | CECRST   | <i>LL_APB1_GRP1_ForceReset</i>     |

| Register | Field | Function                         |
|----------|-------|----------------------------------|
| DAC1RST  |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
|          |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| DAC2RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| I2C1RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| I2C2RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| I2C3RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| PWRRST   |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| SPI2RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| SPI3RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| TIM12RST |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| TIM13RST |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| TIM14RST |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| TIM18RST |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| TIM2RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| TIM3RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| TIM4RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| TIM5RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| TIM6RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |
|          |       | <i>LL_APB1_GRP1_ReleaseReset</i> |
| TIM7RST  |       | <i>LL_APB1_GRP1_ForceReset</i>   |

| Register | Field     | Function                           |
|----------|-----------|------------------------------------|
|          | UART4RST  | <i>LL_APB1_GRP1_ReleaseReset</i>   |
|          |           | <i>LL_APB1_GRP1_ForceReset</i>     |
|          |           | <i>LL_APB1_GRP1_ReleaseReset</i>   |
|          | UART5RST  | <i>LL_APB1_GRP1_ForceReset</i>     |
|          |           | <i>LL_APB1_GRP1_ReleaseReset</i>   |
|          | USART2RST | <i>LL_APB1_GRP1_ForceReset</i>     |
|          |           | <i>LL_APB1_GRP1_ReleaseReset</i>   |
|          | USART3RST | <i>LL_APB1_GRP1_ForceReset</i>     |
|          |           | <i>LL_APB1_GRP1_ReleaseReset</i>   |
|          | USBRST    | <i>LL_APB1_GRP1_ForceReset</i>     |
|          |           | <i>LL_APB1_GRP1_ReleaseReset</i>   |
| APB2ENR  | ADC1EN    | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
|          | HRTIM1EN  | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
|          | SDADC1EN  | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
|          | SDADC2EN  | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
|          | SDADC3EN  | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
|          | SPI1EN    | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
|          | SPI4EN    | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
|          | SYSCFGEN  | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |

| Register | Field     | Function                           |
|----------|-----------|------------------------------------|
| TIM15EN  |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
|          |           | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
| TIM16EN  |           | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
| TIM17EN  |           | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
| TIM19EN  |           | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
| TIM1EN   |           | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
| TIM20EN  |           | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
| TIM8EN   |           | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
| USART1EN |           | <i>LL_APB2_GRP1_DisableClock</i>   |
|          |           | <i>LL_APB2_GRP1_EnableClock</i>    |
|          |           | <i>LL_APB2_GRP1_IsEnabledClock</i> |
| APB2RSTR | ADC1RST   | <i>LL_APB2_GRP1_ForceReset</i>     |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i>   |
|          | HRTIM1RST | <i>LL_APB2_GRP1_ForceReset</i>     |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i>   |
|          | SDADC1RST | <i>LL_APB2_GRP1_ForceReset</i>     |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i>   |
|          | SDADC2RST | <i>LL_APB2_GRP1_ForceReset</i>     |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i>   |
|          | SDADC3RST | <i>LL_APB2_GRP1_ForceReset</i>     |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i>   |
|          | SPI1RST   | <i>LL_APB2_GRP1_ForceReset</i>     |

| Register | Field     | Function                         |
|----------|-----------|----------------------------------|
|          | SPI4RST   | <i>LL_APB2_GRP1_ReleaseReset</i> |
|          |           | <i>LL_APB2_GRP1_ForceReset</i>   |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i> |
|          | SYSCFGRST | <i>LL_APB2_GRP1_ForceReset</i>   |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i> |
|          | TIM15RST  | <i>LL_APB2_GRP1_ForceReset</i>   |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i> |
|          | TIM16RST  | <i>LL_APB2_GRP1_ForceReset</i>   |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i> |
|          | TIM17RST  | <i>LL_APB2_GRP1_ForceReset</i>   |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i> |
|          | TIM19RST  | <i>LL_APB2_GRP1_ForceReset</i>   |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i> |
|          | TIM1RST   | <i>LL_APB2_GRP1_ForceReset</i>   |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i> |
|          | TIM20RST  | <i>LL_APB2_GRP1_ForceReset</i>   |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i> |
|          | TIM8RST   | <i>LL_APB2_GRP1_ForceReset</i>   |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i> |
|          | USART1RST | <i>LL_APB2_GRP1_ForceReset</i>   |
|          |           | <i>LL_APB2_GRP1_ReleaseReset</i> |

## 82.3 COMP

Table 27: Correspondence between COMP registers and COMP low-layer driver functions

| Register | Field         | Function                               |
|----------|---------------|----------------------------------------|
| CSR      | COMPxBLANKING | <i>LL_COMP_GetOutputBlankingSource</i> |
|          |               | <i>LL_COMP_SetOutputBlankingSource</i> |
|          | COMPxEN       | <i>LL_COMP_Disable</i>                 |
|          |               | <i>LL_COMP_Enable</i>                  |
|          |               | <i>LL_COMP_IsEnabled</i>               |
|          | COMPxHYST     | <i>LL_COMP_GetInputHysteresis</i>      |
|          |               | <i>LL_COMP_SetInputHysteresis</i>      |
|          | COMPxLOCK     | <i>LL_COMP_IsLocked</i>                |
|          |               | <i>LL_COMP_Lock</i>                    |
|          | COMPxMODE     | <i>LL_COMP_GetPowerMode</i>            |
|          |               | <i>LL_COMP_SetPowerMode</i>            |

| Register | Field       | Function                           |
|----------|-------------|------------------------------------|
|          | COMPxOUT    | <i>LL_COMP_ReadOutputLevel</i>     |
|          | COMPxOUTSEL | <i>LL_COMP_GetOutputSelection</i>  |
|          |             | <i>LL_COMP_SetOutputSelection</i>  |
|          | COMPxPOL    | <i>LL_COMP_GetOutputPolarity</i>   |
|          |             | <i>LL_COMP_SetOutputPolarity</i>   |
|          | COMPxWNDWEN | <i>LL_COMP_GetCommonWindowMode</i> |
|          |             | <i>LL_COMP_SetCommonWindowMode</i> |
|          | INMSEL      | <i>LL_COMP_ConfigInputs</i>        |
|          |             | <i>LL_COMP_GetInputMinus</i>       |
|          |             | <i>LL_COMP_SetInputMinus</i>       |
|          | NONINSEL    | <i>LL_COMP_ConfigInputs</i>        |
|          |             | <i>LL_COMP_GetInputPlus</i>        |
|          |             | <i>LL_COMP_SetInputPlus</i>        |

## 82.4 CORTEX

Table 28: Correspondence between CORTEX registers and CORTEX low-layer driver functions

| Register  | Field        | Function                       |
|-----------|--------------|--------------------------------|
| MPU_CTRL  | ENABLE       | <i>LL_MPUI_Disable</i>         |
|           |              | <i>LL_MPUI_Enable</i>          |
|           |              | <i>LL_MPUI_IsEnabled</i>       |
| MPU_RASR  | AP           | <i>LL_MPUI_ConfigRegion</i>    |
|           | B            | <i>LL_MPUI_ConfigRegion</i>    |
|           | C            | <i>LL_MPUI_ConfigRegion</i>    |
|           | ENABLE       | <i>LL_MPUI_DisableRegion</i>   |
|           |              | <i>LL_MPUI_EnableRegion</i>    |
|           | S            | <i>LL_MPUI_ConfigRegion</i>    |
|           | SIZE         | <i>LL_MPUI_ConfigRegion</i>    |
| MPU_RBAR  | ADDR         | <i>LL_MPUI_ConfigRegion</i>    |
|           | REGION       | <i>LL_MPUI_ConfigRegion</i>    |
| MPU_RNR   | REGION       | <i>LL_MPUI_ConfigRegion</i>    |
|           |              | <i>LL_MPUI_DisableRegion</i>   |
| SCB_CPUID | ARCHITECTURE | <i>LL_CPUID_GetConstant</i>    |
|           | IMPLEMENTER  | <i>LL_CPUID_GetImplementer</i> |
|           | PARTNO       | <i>LL_CPUID_GetParNo</i>       |
|           | REVISION     | <i>LL_CPUID_GetRevision</i>    |

| Register | Field       | Function                                       |
|----------|-------------|------------------------------------------------|
| SCB_SCR  | VARIANT     | <a href="#">LL_CPUID_GetVariant</a>            |
|          | SEVEONPEND  | <a href="#">LL_LPM_DisableEventOnPend</a>      |
|          |             | <a href="#">LL_LPM_EnableEventOnPend</a>       |
|          | SLEEPDEEP   | <a href="#">LL_LPM_EnableDeepSleep</a>         |
|          |             | <a href="#">LL_LPM_EnableSleep</a>             |
|          | SLEEPONEXIT | <a href="#">LL_LPM_DisableSleepOnExit</a>      |
|          |             | <a href="#">LL_LPM_EnableSleepOnExit</a>       |
|          | MEMFAULTENA | <a href="#">LL_HANDLER_DisableFault</a>        |
|          |             | <a href="#">LL_HANDLER_EnableFault</a>         |
| STK_CTRL | CLKSOURCE   | <a href="#">LL_SYSTICK_GetClkSource</a>        |
|          |             | <a href="#">LL_SYSTICK_SetClkSource</a>        |
|          | COUNTFLAG   | <a href="#">LL_SYSTICK_IsActiveCounterFlag</a> |
|          | TICKINT     | <a href="#">LL_SYSTICK_DisableIT</a>           |
|          |             | <a href="#">LL_SYSTICK_EnableIT</a>            |
|          |             | <a href="#">LL_SYSTICK_IsEnabledIT</a>         |

## 82.5 CRC

Table 29: Correspondence between CRC registers and CRC low-layer driver functions

| Register | Field    | Function                                        |
|----------|----------|-------------------------------------------------|
| CR       | POLYSIZE | <a href="#">LL_CRC_GetPolynomialSize</a>        |
|          |          | <a href="#">LL_CRC_SetPolynomialSize</a>        |
|          | RESET    | <a href="#">LL_CRC_ResetCRCCalculationUnit</a>  |
|          | REV_IN   | <a href="#">LL_CRC_GetInputDataReverseMode</a>  |
|          |          | <a href="#">LL_CRC_SetInputDataReverseMode</a>  |
|          | REV_OUT  | <a href="#">LL_CRC_GetOutputDataReverseMode</a> |
|          |          | <a href="#">LL_CRC_SetOutputDataReverseMode</a> |
|          | DR       | <a href="#">LL_CRC_FeedData16</a>               |
|          |          | <a href="#">LL_CRC_FeedData32</a>               |
|          |          | <a href="#">LL_CRC_FeedData8</a>                |
|          |          | <a href="#">LL_CRC_ReadData16</a>               |
|          |          | <a href="#">LL_CRC_ReadData32</a>               |
|          |          | <a href="#">LL_CRC_ReadData7</a>                |
|          |          | <a href="#">LL_CRC_ReadData8</a>                |
| IDR      | IDR      | <a href="#">LL_CRC_Read_IDR</a>                 |
|          |          | <a href="#">LL_CRC_Write_IDR</a>                |
| INIT     | INIT     | <a href="#">LL_CRC_GetInitialData</a>           |

| Register | Field | Function                        |
|----------|-------|---------------------------------|
| POL      | POL   | <i>LL_CRC_SetInitialData</i>    |
|          |       | <i>LL_CRC_GetPolynomialCoef</i> |
|          |       | <i>LL_CRC_SetPolynomialCoef</i> |

## 82.6 DAC

Table 30: Correspondence between DAC registers and DAC low-layer driver functions

| Register | Field     | Function                               |
|----------|-----------|----------------------------------------|
| CR       | BOFF1     | <i>LL_DAC_GetOutputBuffer</i>          |
|          |           | <i>LL_DAC_SetOutputBuffer</i>          |
|          | BOFF2     | <i>LL_DAC_GetOutputBuffer</i>          |
|          |           | <i>LL_DAC_SetOutputBuffer</i>          |
|          | DMAEN1    | <i>LL_DAC_DisableDMAReq</i>            |
|          |           | <i>LL_DAC_EnableDMAReq</i>             |
|          |           | <i>LL_DAC_IsDMAReqEnabled</i>          |
|          | DMAEN2    | <i>LL_DAC_DisableDMAReq</i>            |
|          |           | <i>LL_DAC_EnableDMAReq</i>             |
|          |           | <i>LL_DAC_IsDMAReqEnabled</i>          |
|          | DMAUDRIE1 | <i>LL_DAC_DisableIT_DMAUDR1</i>        |
|          |           | <i>LL_DAC_EnableIT_DMAUDR1</i>         |
|          |           | <i>LL_DAC_IsEnabledIT_DMAUDR1</i>      |
|          | DMAUDRIE2 | <i>LL_DAC_DisableIT_DMAUDR2</i>        |
|          |           | <i>LL_DAC_EnableIT_DMAUDR2</i>         |
|          |           | <i>LL_DAC_IsEnabledIT_DMAUDR2</i>      |
|          | EN1       | <i>LL_DAC_Disable</i>                  |
|          |           | <i>LL_DAC_Enable</i>                   |
|          |           | <i>LL_DAC_IsEnabled</i>                |
|          | EN2       | <i>LL_DAC_Disable</i>                  |
|          |           | <i>LL_DAC_Enable</i>                   |
|          |           | <i>LL_DAC_IsEnabled</i>                |
|          | MAMP1     | <i>LL_DAC_GetWaveNoiseLFSR</i>         |
|          |           | <i>LL_DAC_GetWaveTriangleAmplitude</i> |
|          |           | <i>LL_DAC_SetWaveNoiseLFSR</i>         |
|          |           | <i>LL_DAC_SetWaveTriangleAmplitude</i> |
|          | MAMP2     | <i>LL_DAC_GetWaveNoiseLFSR</i>         |
|          |           | <i>LL_DAC_GetWaveTriangleAmplitude</i> |
|          |           | <i>LL_DAC_SetWaveNoiseLFSR</i>         |

| Register | Field    | Function                                    |
|----------|----------|---------------------------------------------|
| TEN1     |          | <i>LL_DAC_SetWaveTriangleAmplitude</i>      |
|          |          | <i>LL_DAC_DisableTrigger</i>                |
|          |          | <i>LL_DAC_EnableTrigger</i>                 |
|          |          | <i>LL_DAC_IsTriggerEnabled</i>              |
|          | TEN2     | <i>LL_DAC_DisableTrigger</i>                |
|          |          | <i>LL_DAC_EnableTrigger</i>                 |
|          |          | <i>LL_DAC_IsTriggerEnabled</i>              |
|          | TSEL1    | <i>LL_DAC_GetTriggerSource</i>              |
|          |          | <i>LL_DAC_SetTriggerSource</i>              |
|          | TSEL2    | <i>LL_DAC_GetTriggerSource</i>              |
|          |          | <i>LL_DAC_SetTriggerSource</i>              |
| WAVE1    |          | <i>LL_DAC_GetWaveAutoGeneration</i>         |
|          |          | <i>LL_DAC_SetWaveAutoGeneration</i>         |
|          | WAVE2    | <i>LL_DAC_GetWaveAutoGeneration</i>         |
|          |          | <i>LL_DAC_SetWaveAutoGeneration</i>         |
| DHR12L1  | DACC1DHR | <i>LL_DAC_ConvertData12LeftAligned</i>      |
|          |          | <i>LL_DAC_DMA_GetRegAddr</i>                |
| DHR12L2  | DACC2DHR | <i>LL_DAC_ConvertData12LeftAligned</i>      |
|          |          | <i>LL_DAC_DMA_GetRegAddr</i>                |
| DHR12LD  | DACC1DHR | <i>LL_DAC_ConvertDualData12LeftAligned</i>  |
|          | DACC2DHR | <i>LL_DAC_ConvertDualData12LeftAligned</i>  |
| DHR12R1  | DACC1DHR | <i>LL_DAC_ConvertData12RightAligned</i>     |
|          |          | <i>LL_DAC_DMA_GetRegAddr</i>                |
| DHR12R2  | DACC2DHR | <i>LL_DAC_ConvertData12RightAligned</i>     |
|          |          | <i>LL_DAC_DMA_GetRegAddr</i>                |
| DHR12RD  | DACC1DHR | <i>LL_DAC_ConvertDualData12RightAligned</i> |
|          | DACC2DHR | <i>LL_DAC_ConvertDualData12RightAligned</i> |
| DHR8R1   | DACC1DHR | <i>LL_DAC_ConvertData8RightAligned</i>      |
|          |          | <i>LL_DAC_DMA_GetRegAddr</i>                |
| DHR8R2   | DACC2DHR | <i>LL_DAC_ConvertData8RightAligned</i>      |
|          |          | <i>LL_DAC_DMA_GetRegAddr</i>                |
| DHR8RD   | DACC1DHR | <i>LL_DAC_ConvertDualData8RightAligned</i>  |
|          | DACC2DHR | <i>LL_DAC_ConvertDualData8RightAligned</i>  |
| DOR1     | DACC1DOR | <i>LL_DAC_RetrieveOutputData</i>            |
| DOR2     | DACC2DOR | <i>LL_DAC_RetrieveOutputData</i>            |
| SR       | DMAUDR1  | <i>LL_DAC_ClearFlag_DMAUDR1</i>             |

| Register | Field   | Function                           |
|----------|---------|------------------------------------|
| DMAUDR2  |         | <i>LL_DAC_IsActiveFlag_DMAUDR1</i> |
|          |         | <i>LL_DAC_ClearFlag_DMAUDR2</i>    |
|          |         | <i>LL_DAC_IsActiveFlag_DMAUDR2</i> |
| SWTRIGR  | SWTRIG1 | <i>LL_DAC_TrigSWConversion</i>     |
|          | SWTRIG2 | <i>LL_DAC_TrigSWConversion</i>     |

## 82.7 DMA

Table 31: Correspondence between DMA registers and DMA low-layer driver functions

| Register | Field   | Function                               |
|----------|---------|----------------------------------------|
| CCR      | CIRC    | <i>LL_DMA_ConfigTransfer</i>           |
|          |         | <i>LL_DMA_GetMode</i>                  |
|          |         | <i>LL_DMA_SetMode</i>                  |
|          | DIR     | <i>LL_DMA_ConfigTransfer</i>           |
|          |         | <i>LL_DMA_GetDataTransferDirection</i> |
|          |         | <i>LL_DMA_SetDataTransferDirection</i> |
|          | EN      | <i>LL_DMA_DisableChannel</i>           |
|          |         | <i>LL_DMA_EnableChannel</i>            |
|          |         | <i>LL_DMA_IsEnabledChannel</i>         |
|          | HTIE    | <i>LL_DMA_DisableIT_HT</i>             |
|          |         | <i>LL_DMA_EnableIT_HT</i>              |
|          |         | <i>LL_DMA_IsEnabledIT_HT</i>           |
|          | MEM2MEM | <i>LL_DMA_ConfigTransfer</i>           |
|          |         | <i>LL_DMA_GetDataTransferDirection</i> |
|          |         | <i>LL_DMA_SetDataTransferDirection</i> |
|          | MINC    | <i>LL_DMA_ConfigTransfer</i>           |
|          |         | <i>LL_DMA_GetMemoryIncMode</i>         |
|          |         | <i>LL_DMA_SetMemoryIncMode</i>         |
|          | MSIZE   | <i>LL_DMA_ConfigTransfer</i>           |
|          |         | <i>LL_DMA_GetMemorySize</i>            |
|          |         | <i>LL_DMA_SetMemorySize</i>            |
|          | PINC    | <i>LL_DMA_ConfigTransfer</i>           |
|          |         | <i>LL_DMA_GetPeriphIncMode</i>         |
|          |         | <i>LL_DMA_SetPeriphIncMode</i>         |
|          | PL      | <i>LL_DMA_ConfigTransfer</i>           |
|          |         | <i>LL_DMA_GetChannelPriorityLevel</i>  |
|          |         | <i>LL_DMA_SetChannelPriorityLevel</i>  |

| Register | Field  | Function                                       |
|----------|--------|------------------------------------------------|
|          | PSIZE  | <a href="#"><i>LL_DMA_ConfigTransfer</i></a>   |
|          |        | <a href="#"><i>LL_DMA_GetPeriphSize</i></a>    |
|          |        | <a href="#"><i>LL_DMA_SetPeriphSize</i></a>    |
|          | TCIE   | <a href="#"><i>LL_DMA_DisableIT_TC</i></a>     |
|          |        | <a href="#"><i>LL_DMA_EnableIT_TC</i></a>      |
|          |        | <a href="#"><i>LL_DMA_IsEnabledIT_TC</i></a>   |
|          | TEIE   | <a href="#"><i>LL_DMA_DisableIT_TE</i></a>     |
|          |        | <a href="#"><i>LL_DMA_EnableIT_TE</i></a>      |
|          |        | <a href="#"><i>LL_DMA_IsEnabledIT_TE</i></a>   |
|          | CMAR   | <a href="#"><i>LL_DMA_ConfigAddresses</i></a>  |
|          |        | <a href="#"><i>LL_DMA_GetM2MDstAddress</i></a> |
|          |        | <a href="#"><i>LL_DMA_GetMemoryAddress</i></a> |
|          |        | <a href="#"><i>LL_DMA_SetM2MDstAddress</i></a> |
|          |        | <a href="#"><i>LL_DMA_SetMemoryAddress</i></a> |
|          | CNDTR  | <a href="#"><i>LL_DMA_GetDataLength</i></a>    |
|          |        | <a href="#"><i>LL_DMA_SetDataLength</i></a>    |
|          | CPAR   | <a href="#"><i>LL_DMA_ConfigAddresses</i></a>  |
|          |        | <a href="#"><i>LL_DMA_GetM2MSrcAddress</i></a> |
|          |        | <a href="#"><i>LL_DMA_GetPeriphAddress</i></a> |
|          |        | <a href="#"><i>LL_DMA_SetM2MSrcAddress</i></a> |
|          |        | <a href="#"><i>LL_DMA_SetPeriphAddress</i></a> |
| IFCR     | CGIF1  | <a href="#"><i>LL_DMA_ClearFlag_GI1</i></a>    |
|          | CGIF2  | <a href="#"><i>LL_DMA_ClearFlag_GI2</i></a>    |
|          | CGIF3  | <a href="#"><i>LL_DMA_ClearFlag_GI3</i></a>    |
|          | CGIF4  | <a href="#"><i>LL_DMA_ClearFlag_GI4</i></a>    |
|          | CGIF5  | <a href="#"><i>LL_DMA_ClearFlag_GI5</i></a>    |
|          | CGIF6  | <a href="#"><i>LL_DMA_ClearFlag_GI6</i></a>    |
|          | CGIF7  | <a href="#"><i>LL_DMA_ClearFlag_GI7</i></a>    |
|          | CHTIF1 | <a href="#"><i>LL_DMA_ClearFlag_HT1</i></a>    |
|          | CHTIF2 | <a href="#"><i>LL_DMA_ClearFlag_HT2</i></a>    |
|          | CHTIF3 | <a href="#"><i>LL_DMA_ClearFlag_HT3</i></a>    |
|          | CHTIF4 | <a href="#"><i>LL_DMA_ClearFlag_HT4</i></a>    |
|          | CHTIF5 | <a href="#"><i>LL_DMA_ClearFlag_HT5</i></a>    |
|          | CHTIF6 | <a href="#"><i>LL_DMA_ClearFlag_HT6</i></a>    |
|          | CHTIF7 | <a href="#"><i>LL_DMA_ClearFlag_HT7</i></a>    |
|          | CTCIF1 | <a href="#"><i>LL_DMA_ClearFlag_TC1</i></a>    |

| Register | Field  | Function                       |
|----------|--------|--------------------------------|
| ISR      | CTCIF2 | <i>LL_DMA_ClearFlag_TC2</i>    |
|          | CTCIF3 | <i>LL_DMA_ClearFlag_TC3</i>    |
|          | CTCIF4 | <i>LL_DMA_ClearFlag_TC4</i>    |
|          | CTCIF5 | <i>LL_DMA_ClearFlag_TC5</i>    |
|          | CTCIF6 | <i>LL_DMA_ClearFlag_TC6</i>    |
|          | CTCIF7 | <i>LL_DMA_ClearFlag_TC7</i>    |
|          | CTEIF1 | <i>LL_DMA_ClearFlag_TE1</i>    |
|          | CTEIF2 | <i>LL_DMA_ClearFlag_TE2</i>    |
|          | CTEIF3 | <i>LL_DMA_ClearFlag_TE3</i>    |
|          | CTEIF4 | <i>LL_DMA_ClearFlag_TE4</i>    |
|          | CTEIF5 | <i>LL_DMA_ClearFlag_TE5</i>    |
|          | CTEIF6 | <i>LL_DMA_ClearFlag_TE6</i>    |
|          | CTEIF7 | <i>LL_DMA_ClearFlag_TE7</i>    |
|          | GIF1   | <i>LL_DMA_IsActiveFlag_GI1</i> |
| ISR      | GIF2   | <i>LL_DMA_IsActiveFlag_GI2</i> |
|          | GIF3   | <i>LL_DMA_IsActiveFlag_GI3</i> |
|          | GIF4   | <i>LL_DMA_IsActiveFlag_GI4</i> |
|          | GIF5   | <i>LL_DMA_IsActiveFlag_GI5</i> |
|          | GIF6   | <i>LL_DMA_IsActiveFlag_GI6</i> |
|          | GIF7   | <i>LL_DMA_IsActiveFlag_GI7</i> |
|          | HTIF1  | <i>LL_DMA_IsActiveFlag_HT1</i> |
|          | HTIF2  | <i>LL_DMA_IsActiveFlag_HT2</i> |
|          | HTIF3  | <i>LL_DMA_IsActiveFlag_HT3</i> |
|          | HTIF4  | <i>LL_DMA_IsActiveFlag_HT4</i> |
|          | HTIF5  | <i>LL_DMA_IsActiveFlag_HT5</i> |
|          | HTIF6  | <i>LL_DMA_IsActiveFlag_HT6</i> |
|          | HTIF7  | <i>LL_DMA_IsActiveFlag_HT7</i> |
|          | TCIF1  | <i>LL_DMA_IsActiveFlag_TC1</i> |
| ISR      | TCIF2  | <i>LL_DMA_IsActiveFlag_TC2</i> |
|          | TCIF3  | <i>LL_DMA_IsActiveFlag_TC3</i> |
|          | TCIF4  | <i>LL_DMA_IsActiveFlag_TC4</i> |
|          | TCIF5  | <i>LL_DMA_IsActiveFlag_TC5</i> |
|          | TCIF6  | <i>LL_DMA_IsActiveFlag_TC6</i> |
|          | TCIF7  | <i>LL_DMA_IsActiveFlag_TC7</i> |
|          | TEIF1  | <i>LL_DMA_IsActiveFlag_TE1</i> |
| ISR      | TEIF2  | <i>LL_DMA_IsActiveFlag_TE2</i> |

| Register | Field | Function                                |
|----------|-------|-----------------------------------------|
|          | TEIF3 | <a href="#">LL_DMA_IsActiveFlag_TE3</a> |
|          | TEIF4 | <a href="#">LL_DMA_IsActiveFlag_TE4</a> |
|          | TEIF5 | <a href="#">LL_DMA_IsActiveFlag_TE5</a> |
|          | TEIF6 | <a href="#">LL_DMA_IsActiveFlag_TE6</a> |
|          | TEIF7 | <a href="#">LL_DMA_IsActiveFlag_TE7</a> |

## 82.8 EXTI

Table 32: Correspondence between EXTI registers and EXTI low-layer driver functions

| Register | Field | Function                                           |
|----------|-------|----------------------------------------------------|
| EMR      | EMx   | <a href="#">LL_EXTI_DisableEvent_0_31</a>          |
|          |       | <a href="#">LL_EXTI_EnableEvent_0_31</a>           |
|          |       | <a href="#">LL_EXTI_IsEnabledEvent_0_31</a>        |
| EMR2     | EMx   | <a href="#">LL_EXTI_DisableEvent_32_63</a>         |
|          |       | <a href="#">LL_EXTI_EnableEvent_32_63</a>          |
|          |       | <a href="#">LL_EXTI_IsEnabledEvent_32_63</a>       |
| FTSR     | FTx   | <a href="#">LL_EXTI_DisableFallingTrig_0_31</a>    |
|          |       | <a href="#">LL_EXTI_EnableFallingTrig_0_31</a>     |
|          |       | <a href="#">LL_EXTI_IsEnabledFallingTrig_0_31</a>  |
| FTSR2    | FTx   | <a href="#">LL_EXTI_DisableFallingTrig_32_63</a>   |
|          |       | <a href="#">LL_EXTI_EnableFallingTrig_32_63</a>    |
|          |       | <a href="#">LL_EXTI_IsEnabledFallingTrig_32_63</a> |
| IMR      | IMx   | <a href="#">LL_EXTI_DisableIT_0_31</a>             |
|          |       | <a href="#">LL_EXTI_EnableIT_0_31</a>              |
|          |       | <a href="#">LL_EXTI_IsEnabledIT_0_31</a>           |
| IMR2     | IMx   | <a href="#">LL_EXTI_DisableIT_32_63</a>            |
|          |       | <a href="#">LL_EXTI_EnableIT_32_63</a>             |
|          |       | <a href="#">LL_EXTI_IsEnabledIT_32_63</a>          |
| PR       | PIFx  | <a href="#">LL_EXTI_ClearFlag_0_31</a>             |
|          |       | <a href="#">LL_EXTI_IsActiveFlag_0_31</a>          |
|          |       | <a href="#">LL_EXTI_ReadFlag_0_31</a>              |
| PR2      | PIFx  | <a href="#">LL_EXTI_ClearFlag_32_63</a>            |
|          |       | <a href="#">LL_EXTI_IsActiveFlag_32_63</a>         |
|          |       | <a href="#">LL_EXTI_ReadFlag_32_63</a>             |
| RTSR     | RTx   | <a href="#">LL_EXTI_DisableRisingTrig_0_31</a>     |
|          |       | <a href="#">LL_EXTI_EnableRisingTrig_0_31</a>      |
|          |       | <a href="#">LL_EXTI_IsEnabledRisingTrig_0_31</a>   |

| Register | Field | Function                                                 |
|----------|-------|----------------------------------------------------------|
| RTSR2    | RTx   | <a href="#"><i>LL_EXTI_DisableRisingTrig_32_63</i></a>   |
|          |       | <a href="#"><i>LL_EXTI_EnableRisingTrig_32_63</i></a>    |
|          |       | <a href="#"><i>LL_EXTI_IsEnabledRisingTrig_32_63</i></a> |
| SWIER    | SWIx  | <a href="#"><i>LL_EXTI_GenerateSWI_0_31</i></a>          |
| SWIER2   | SWIx  | <a href="#"><i>LL_EXTI_GenerateSWI_32_63</i></a>         |

## 82.9 GPIO

Table 33: Correspondence between GPIO registers and GPIO low-layer driver functions

| Register | Field   | Function                                        |
|----------|---------|-------------------------------------------------|
| AFRH     | AFSELy  | <a href="#"><i>LL_GPIO_GetAPPin_8_15</i></a>    |
|          |         | <a href="#"><i>LL_GPIO_SetAPPin_8_15</i></a>    |
| AFRL     | AFSELy  | <a href="#"><i>LL_GPIO_GetAPPin_0_7</i></a>     |
|          |         | <a href="#"><i>LL_GPIO_SetAPPin_0_7</i></a>     |
| BRR      | BRy     | <a href="#"><i>LL_GPIO_ResetOutputPin</i></a>   |
| BSRR     | BSy     | <a href="#"><i>LL_GPIO_SetOutputPin</i></a>     |
| IDR      | IDy     | <a href="#"><i>LL_GPIO_IsInputPinSet</i></a>    |
|          |         | <a href="#"><i>LL_GPIO_ReadInputPort</i></a>    |
| LCKR     | LCKK    | <a href="#"><i>LL_GPIO_IsAnyPinLocked</i></a>   |
|          |         | <a href="#"><i>LL_GPIO_LockPin</i></a>          |
|          | LCKy    | <a href="#"><i>LL_GPIO_IsPinLocked</i></a>      |
| MODER    | MODEy   | <a href="#"><i>LL_GPIO_GetPinMode</i></a>       |
|          |         | <a href="#"><i>LL_GPIO_SetPinMode</i></a>       |
| ODR      | ODy     | <a href="#"><i>LL_GPIO_IsOutputPinSet</i></a>   |
|          |         | <a href="#"><i>LL_GPIO_ReadOutputPort</i></a>   |
|          |         | <a href="#"><i>LL_GPIO_TogglePin</i></a>        |
|          |         | <a href="#"><i>LL_GPIO_WriteOutputPort</i></a>  |
| OSPEEDR  | OSPEEDy | <a href="#"><i>LL_GPIO_GetPinSpeed</i></a>      |
|          |         | <a href="#"><i>LL_GPIO_SetPinSpeed</i></a>      |
| OTYPER   | OTy     | <a href="#"><i>LL_GPIO_GetPinOutputType</i></a> |
|          |         | <a href="#"><i>LL_GPIO_SetPinOutputType</i></a> |
| PUPDR    | PUPDy   | <a href="#"><i>LL_GPIO_GetPinPull</i></a>       |
|          |         | <a href="#"><i>LL_GPIO_SetPinPull</i></a>       |

**82.10 I2C****Table 34: Correspondence between I2C registers and I2C low-layer driver functions**

| Register | Field     | Function                               |
|----------|-----------|----------------------------------------|
| CR1      | ADDRIE    | <i>LL_I2C_DisableIT_ADDR</i>           |
|          |           | <i>LL_I2C_EnableIT_ADDR</i>            |
|          |           | <i>LL_I2C_IsEnabledIT_ADDR</i>         |
|          | ALERTEN   | <i>LL_I2C_DisableSMBusAlert</i>        |
|          |           | <i>LL_I2C_EnableSMBusAlert</i>         |
|          |           | <i>LL_I2C_IsEnabledSMBusAlert</i>      |
|          | ANFOFF    | <i>LL_I2C_ConfigFilters</i>            |
|          |           | <i>LL_I2C_DisableAnalogFilter</i>      |
|          |           | <i>LL_I2C_EnableAnalogFilter</i>       |
|          |           | <i>LL_I2C_IsEnabledAnalogFilter</i>    |
|          | DNF       | <i>LL_I2C_ConfigFilters</i>            |
|          |           | <i>LL_I2C_GetDigitalFilter</i>         |
|          |           | <i>LL_I2C_SetDigitalFilter</i>         |
|          | ERRIE     | <i>LL_I2C_DisableIT_ERR</i>            |
|          |           | <i>LL_I2C_EnableIT_ERR</i>             |
|          |           | <i>LL_I2C_IsEnabledIT_ERR</i>          |
|          | GCEN      | <i>LL_I2C_DisableGeneralCall</i>       |
|          |           | <i>LL_I2C_EnableGeneralCall</i>        |
|          |           | <i>LL_I2C_IsEnabledGeneralCall</i>     |
|          | NACKIE    | <i>LL_I2C_DisableIT_NACK</i>           |
|          |           | <i>LL_I2C_EnableIT_NACK</i>            |
|          |           | <i>LL_I2C_IsEnabledIT_NACK</i>         |
|          | NOSTRETCH | <i>LL_I2C_DisableClockStretching</i>   |
|          |           | <i>LL_I2C_EnableClockStretching</i>    |
|          |           | <i>LL_I2C_IsEnabledClockStretching</i> |
|          | PE        | <i>LL_I2C_Disable</i>                  |
|          |           | <i>LL_I2C_Enable</i>                   |
|          |           | <i>LL_I2C_IsEnabled</i>                |
|          | PECEN     | <i>LL_I2C_DisableSMBusPEC</i>          |
|          |           | <i>LL_I2C_EnableSMBusPEC</i>           |
|          |           | <i>LL_I2C_IsEnabledSMBusPEC</i>        |
|          | RXDMAEN   | <i>LL_I2C_DisableDMAReq_RX</i>         |
|          |           | <i>LL_I2C_EnableDMAReq_RX</i>          |
|          |           | <i>LL_I2C_IsEnabledDMAReq_RX</i>       |

| Register | Field   | Function                                |
|----------|---------|-----------------------------------------|
|          | RXIE    | <i>LL_I2C_DisableIT_RX</i>              |
|          |         | <i>LL_I2C_EnableIT_RX</i>               |
|          |         | <i>LL_I2C_IsEnabledIT_RX</i>            |
|          | SBC     | <i>LL_I2C_DisableSlaveByteControl</i>   |
|          |         | <i>LL_I2C_EnableSlaveByteControl</i>    |
|          |         | <i>LL_I2C_IsEnabledSlaveByteControl</i> |
|          | SMBDEN  | <i>LL_I2C_GetMode</i>                   |
|          |         | <i>LL_I2C_SetMode</i>                   |
|          | SMBHEN  | <i>LL_I2C_GetMode</i>                   |
|          |         | <i>LL_I2C_SetMode</i>                   |
|          | STOPIE  | <i>LL_I2C_DisableIT_STOP</i>            |
|          |         | <i>LL_I2C_EnableIT_STOP</i>             |
|          |         | <i>LL_I2C_IsEnabledIT_STOP</i>          |
|          | TCIE    | <i>LL_I2C_DisableIT_TC</i>              |
|          |         | <i>LL_I2C_EnableIT_TC</i>               |
|          |         | <i>LL_I2C_IsEnabledIT_TC</i>            |
|          | TXDMAEN | <i>LL_I2C_DisableDMAReq_TX</i>          |
|          |         | <i>LL_I2C_EnableDMAReq_TX</i>           |
|          |         | <i>LL_I2C_IsEnabledDMAReq_TX</i>        |
|          | TXIE    | <i>LL_I2C_DisableIT_TX</i>              |
|          |         | <i>LL_I2C_EnableIT_TX</i>               |
|          |         | <i>LL_I2C_IsEnabledIT_TX</i>            |
|          | WUPEN   | <i>LL_I2C_DisableWakeUpFromStop</i>     |
|          |         | <i>LL_I2C_EnableWakeUpFromStop</i>      |
|          |         | <i>LL_I2C_IsEnabledWakeUpFromStop</i>   |
|          | ADD10   | <i>LL_I2C_GetMasterAddressingMode</i>   |
|          |         | <i>LL_I2C_HandleTransfer</i>            |
|          |         | <i>LL_I2C_SetMasterAddressingMode</i>   |
|          | AUTOEND | <i>LL_I2C_DisableAutoEndMode</i>        |
|          |         | <i>LL_I2C_EnableAutoEndMode</i>         |
|          |         | <i>LL_I2C_HandleTransfer</i>            |
|          |         | <i>LL_I2C_IsEnabledAutoEndMode</i>      |
|          | HEAD10R | <i>LL_I2C_DisableAuto10BitRead</i>      |
|          |         | <i>LL_I2C_EnableAuto10BitRead</i>       |
|          |         | <i>LL_I2C_HandleTransfer</i>            |
|          |         | <i>LL_I2C_IsEnabledAuto10BitRead</i>    |

| Register | Field    | Function                                |
|----------|----------|-----------------------------------------|
|          | NACK     | <i>LL_I2C_AcknowledgeNextData</i>       |
|          | NBYTES   | <i>LL_I2C_GetTransferSize</i>           |
|          |          | <i>LL_I2C_HandleTransfer</i>            |
|          |          | <i>LL_I2C_SetTransferSize</i>           |
|          | PECBYTE  | <i>LL_I2C_EnableSMBusPECCCompare</i>    |
|          |          | <i>LL_I2C_IsEnabledSMBusPECCCompare</i> |
|          | RD_WRN   | <i>LL_I2C_GetTransferRequest</i>        |
|          |          | <i>LL_I2C_HandleTransfer</i>            |
|          |          | <i>LL_I2C_SetTransferRequest</i>        |
|          | RELOAD   | <i>LL_I2C_DisableReloadMode</i>         |
|          |          | <i>LL_I2C_EnableReloadMode</i>          |
|          |          | <i>LL_I2C_HandleTransfer</i>            |
|          |          | <i>LL_I2C_IsEnabledReloadMode</i>       |
|          | SADD     | <i>LL_I2C_GetSlaveAddr</i>              |
|          |          | <i>LL_I2C_HandleTransfer</i>            |
|          |          | <i>LL_I2C_SetSlaveAddr</i>              |
|          | START    | <i>LL_I2C_GenerateStartCondition</i>    |
|          |          | <i>LL_I2C_HandleTransfer</i>            |
|          | STOP     | <i>LL_I2C_GenerateStopCondition</i>     |
|          |          | <i>LL_I2C_HandleTransfer</i>            |
| ICR      | ADDRCF   | <i>LL_I2C_ClearFlag_ADDR</i>            |
|          | ALERTCF  | <i>LL_I2C_ClearSMBusFlag_ALERT</i>      |
|          | ARLOCF   | <i>LL_I2C_ClearFlag_ARLO</i>            |
|          | BERRCF   | <i>LL_I2C_ClearFlag_BERR</i>            |
|          | NACKCF   | <i>LL_I2C_ClearFlag_NACK</i>            |
|          | OVRCF    | <i>LL_I2C_ClearFlag_OVR</i>             |
|          | PECCF    | <i>LL_I2C_ClearSMBusFlag_PECERR</i>     |
|          | STOPCF   | <i>LL_I2C_ClearFlag_STOP</i>            |
|          | TIMOUTCF | <i>LL_I2C_ClearSMBusFlag_TIMEOUT</i>    |
| ISR      | ADDCODE  | <i>LL_I2C_GetAddressMatchCode</i>       |
|          | ADDR     | <i>LL_I2C_IsActiveFlag_ADDR</i>         |
|          | ALERT    | <i>LL_I2C_IsActiveSMBusFlag_ALERT</i>   |
|          | ARLO     | <i>LL_I2C_IsActiveFlag_ARLO</i>         |
|          | BERR     | <i>LL_I2C_IsActiveFlag_BERR</i>         |
|          | BUSY     | <i>LL_I2C_IsActiveFlag_BUSY</i>         |
|          | DIR      | <i>LL_I2C_GetTransferDirection</i>      |

| Register | Field    | Function                                |
|----------|----------|-----------------------------------------|
|          | NACKF    | <i>LL_I2C_IsActiveFlag_NACK</i>         |
|          | OVR      | <i>LL_I2C_IsActiveFlag_OVR</i>          |
|          | PECERR   | <i>LL_I2C_IsActiveSMBusFlag_PECERR</i>  |
|          | RXNE     | <i>LL_I2C_IsActiveFlag_RXNE</i>         |
|          | STOPF    | <i>LL_I2C_IsActiveFlag_STOP</i>         |
|          | TC       | <i>LL_I2C_IsActiveFlag_TC</i>           |
|          | TCR      | <i>LL_I2C_IsActiveFlag_TCR</i>          |
|          | TIMEOUT  | <i>LL_I2C_IsActiveSMBusFlag_TIMEOUT</i> |
|          | TXE      | <i>LL_I2C_ClearFlag_TXE</i>             |
|          |          | <i>LL_I2C_IsActiveFlag_TXE</i>          |
|          | TXIS     | <i>LL_I2C_IsActiveFlag_TXIS</i>         |
| OAR1     | OA1      | <i>LL_I2C_SetOwnAddress1</i>            |
|          | OA1EN    | <i>LL_I2C_DisableOwnAddress1</i>        |
|          |          | <i>LL_I2C_EnableOwnAddress1</i>         |
|          |          | <i>LL_I2C_IsEnabledOwnAddress1</i>      |
|          | OA1MODE  | <i>LL_I2C_SetOwnAddress1</i>            |
| OAR2     | OA2      | <i>LL_I2C_SetOwnAddress2</i>            |
|          | OA2EN    | <i>LL_I2C_DisableOwnAddress2</i>        |
|          |          | <i>LL_I2C_EnableOwnAddress2</i>         |
|          |          | <i>LL_I2C_IsEnabledOwnAddress2</i>      |
|          | OA2MSK   | <i>LL_I2C_SetOwnAddress2</i>            |
| PECR     | PEC      | <i>LL_I2C_GetSMBusPEC</i>               |
| RXDR     | RXDATA   | <i>LL_I2C_DMA_GetRegAddr</i>            |
|          |          | <i>LL_I2C_ReceiveData8</i>              |
| TIMEOUTR | TEXTEN   | <i>LL_I2C_DisableSMBusTimeout</i>       |
|          |          | <i>LL_I2C_EnableSMBusTimeout</i>        |
|          |          | <i>LL_I2C_IsEnabledSMBusTimeout</i>     |
|          | TIDLE    | <i>LL_I2C_ConfigSMBusTimeout</i>        |
|          |          | <i>LL_I2C_GetSMBusTimeoutAMode</i>      |
|          |          | <i>LL_I2C_SetSMBusTimeoutAMode</i>      |
|          | TIMEOUTA | <i>LL_I2C_ConfigSMBusTimeout</i>        |
|          |          | <i>LL_I2C_GetSMBusTimeoutA</i>          |
|          |          | <i>LL_I2C_SetSMBusTimeoutA</i>          |
|          | TIMEOUTB | <i>LL_I2C_ConfigSMBusTimeout</i>        |
|          |          | <i>LL_I2C_GetSMBusTimeoutB</i>          |
|          |          | <i>LL_I2C_SetSMBusTimeoutB</i>          |

| Register | Field    | Function                                     |
|----------|----------|----------------------------------------------|
|          | TIMOUTEN | <a href="#">LL_I2C_DisableSMBusTimeout</a>   |
|          |          | <a href="#">LL_I2C_EnableSMBusTimeout</a>    |
|          |          | <a href="#">LL_I2C_IsEnabledSMBusTimeout</a> |
| TIMINGR  | PRESC    | <a href="#">LL_I2C_GetTimingPrescaler</a>    |
|          | SCLDEL   | <a href="#">LL_I2C_GetDataSetupTime</a>      |
|          | SCLH     | <a href="#">LL_I2C_GetClockHighPeriod</a>    |
|          | SCLL     | <a href="#">LL_I2C_GetClockLowPeriod</a>     |
|          | SDADEL   | <a href="#">LL_I2C_GetDataHoldTime</a>       |
|          | TIMINGR  | <a href="#">LL_I2C_SetTiming</a>             |
| TXDR     | TXDATA   | <a href="#">LL_I2C_DMA_GetRegAddr</a>        |
|          |          | <a href="#">LL_I2C_TransmitData8</a>         |

## 82.11 I2S

Table 35: Correspondence between I2S registers and I2S low-layer driver functions

| Register | Field   | Function                                  |
|----------|---------|-------------------------------------------|
| CR2      | ERRIE   | <a href="#">LL_I2S_DisableIT_ERR</a>      |
|          |         | <a href="#">LL_I2S_EnableIT_ERR</a>       |
|          |         | <a href="#">LL_I2S_IsEnabledIT_ERR</a>    |
|          | RXDMAEN | <a href="#">LL_I2S_DisableDMAReq_RX</a>   |
|          |         | <a href="#">LL_I2S_EnableDMAReq_RX</a>    |
|          |         | <a href="#">LL_I2S_IsEnabledDMAReq_RX</a> |
|          | RXNEIE  | <a href="#">LL_I2S_DisableIT_RXNE</a>     |
|          |         | <a href="#">LL_I2S_EnableIT_RXNE</a>      |
|          |         | <a href="#">LL_I2S_IsEnabledIT_RXNE</a>   |
|          | TXDMAEN | <a href="#">LL_I2S_DisableDMAReq_TX</a>   |
|          |         | <a href="#">LL_I2S_EnableDMAReq_TX</a>    |
|          |         | <a href="#">LL_I2S_IsEnabledDMAReq_TX</a> |
|          | TXEIE   | <a href="#">LL_I2S_DisableIT_TXE</a>      |
|          |         | <a href="#">LL_I2S_EnableIT_TXE</a>       |
|          |         | <a href="#">LL_I2S_IsEnabledIT_TXE</a>    |
| DR       | DR      | <a href="#">LL_I2S_ReceiveData16</a>      |
|          |         | <a href="#">LL_I2S_TransmitData16</a>     |
| I2SCFGR  | CHLEN   | <a href="#">LL_I2S_GetDataFormat</a>      |
|          |         | <a href="#">LL_I2S_SetDataFormat</a>      |
|          | CKPOL   | <a href="#">LL_I2S_GetClockPolarity</a>   |
|          |         | <a href="#">LL_I2S_SetClockPolarity</a>   |

| Register | Field   | Function                                                             |
|----------|---------|----------------------------------------------------------------------|
|          | DATLEN  | <i>LL_I2S_GetDataFormat</i><br><i>LL_I2S_SetDataFormat</i>           |
|          | I2SCFG  | <i>LL_I2S_GetTransferMode</i>                                        |
|          |         | <i>LL_I2S_SetTransferMode</i>                                        |
|          | I2SE    | <i>LL_I2S_Disable</i>                                                |
|          |         | <i>LL_I2S_Enable</i>                                                 |
|          |         | <i>LL_I2S_IsEnabled</i>                                              |
|          | I2SMOD  | <i>LL_I2S_Enable</i>                                                 |
|          | I2SSTD  | <i>LL_I2S_GetStandard</i>                                            |
|          |         | <i>LL_I2S_SetStandard</i>                                            |
|          | PCMSYNC | <i>LL_I2S_GetStandard</i>                                            |
|          |         | <i>LL_I2S_SetStandard</i>                                            |
| I2SPR    | I2SDIV  | <i>LL_I2S_GetPrescalerLinear</i><br><i>LL_I2S_SetPrescalerLinear</i> |
|          |         | <i>LL_I2S_DisableMasterClock</i>                                     |
|          | MCKOE   | <i>LL_I2S_EnableMasterClock</i>                                      |
|          |         | <i>LL_I2S_IsEnabledMasterClock</i>                                   |
|          | ODD     | <i>LL_I2S_GetPrescalerParity</i>                                     |
|          |         | <i>LL_I2S_SetPrescalerParity</i>                                     |
| SR       | BSY     | <i>LL_I2S_IsActiveFlag_BSY</i>                                       |
|          | CHSIDE  | <i>LL_I2S_IsActiveFlag_CHSIDE</i>                                    |
|          | FRE     | <i>LL_I2S_ClearFlag_FRE</i><br><i>LL_I2S_IsActiveFlag_FRE</i>        |
|          |         | <i>LL_I2S_ClearFlag_OVR</i><br><i>LL_I2S_IsActiveFlag_OVR</i>        |
|          | RXNE    | <i>LL_I2S_IsActiveFlag_RXNE</i>                                      |
|          | TXE     | <i>LL_I2S_IsActiveFlag_TXE</i>                                       |
|          | UDR     | <i>LL_I2S_ClearFlag_UDR</i>                                          |
|          |         | <i>LL_I2S_IsActiveFlag_UDR</i>                                       |

## 82.12 IWDG

Table 36: Correspondence between IWDG registers and IWDG low-layer driver functions

| Register | Field | Function                          |
|----------|-------|-----------------------------------|
| KR       | KEY   | <i>LL_IWDG_DisableWriteAccess</i> |
|          |       | <i>LL_IWDG_Enable</i>             |
|          |       | <i>LL_IWDG_EnableWriteAccess</i>  |

| Register | Field | Function                        |
|----------|-------|---------------------------------|
| PR       | PR    | <i>LL_IWDG_ReloadCounter</i>    |
|          |       | <i>LL_IWDG_GetPrescaler</i>     |
|          |       | <i>LL_IWDG_SetPrescaler</i>     |
| RLR      | RL    | <i>LL_IWDG_GetReloadCounter</i> |
|          |       | <i>LL_IWDG_SetReloadCounter</i> |
| SR       | PVU   | <i>LL_IWDG_IsActiveFlag_PVU</i> |
|          |       | <i>LL_IWDG_IsReady</i>          |
|          | RVU   | <i>LL_IWDG_IsActiveFlag_RVU</i> |
|          |       | <i>LL_IWDG_IsReady</i>          |
|          | WVU   | <i>LL_IWDG_IsActiveFlag_WVU</i> |
|          |       | <i>LL_IWDG_IsReady</i>          |
| WINR     | WIN   | <i>LL_IWDG_GetWindow</i>        |
|          |       | <i>LL_IWDG_SetWindow</i>        |

## 82.13 OPAMP

Table 37: Correspondence between OPAMP registers and OPAMP low-layer driver functions

| Register | Field       | Function                                 |
|----------|-------------|------------------------------------------|
| CSR      | CALON       | <i>LL_OPAMP_GetMode</i>                  |
|          |             | <i>LL_OPAMP_SetMode</i>                  |
|          | CALSEL      | <i>LL_OPAMP_GetCalibrationSelection</i>  |
|          |             | <i>LL_OPAMP_SetCalibrationSelection</i>  |
|          | LOCK        | <i>LL_OPAMP_IsLocked</i>                 |
|          |             | <i>LL_OPAMP_Lock</i>                     |
|          | OPAMPXEN    | <i>LL_OPAMP_Disable</i>                  |
|          |             | <i>LL_OPAMP_Enable</i>                   |
|          |             | <i>LL_OPAMP_IsEnabled</i>                |
|          | OUTCAL      | <i>LL_OPAMP_IsCalibrationOutputSet</i>   |
|          | PGGAIN      | <i>LL_OPAMP_GetPGAGain</i>               |
|          |             | <i>LL_OPAMP_SetPGAGain</i>               |
|          | TCMEN       | <i>LL_OPAMP_GetInputsMuxMode</i>         |
|          |             | <i>LL_OPAMP_SetInputsMuxMode</i>         |
|          | TRIMOFFSETN | <i>LL_OPAMP_GetTrimmingValue</i>         |
|          |             | <i>LL_OPAMP_SetTrimmingValue</i>         |
|          | TRIMOFFSETP | <i>LL_OPAMP_GetTrimmingValue</i>         |
|          |             | <i>LL_OPAMP_SetTrimmingValue</i>         |
|          | TSTREF      | <i>LL_OPAMP_GetCalibrationVrefOutput</i> |

| Register | Field    | Function                                      |
|----------|----------|-----------------------------------------------|
|          | USERTRIM | <i>LL_OPAMP_SetCalibrationVrefOutput</i>      |
|          |          | <i>LL_OPAMP_GetTrimmingMode</i>               |
|          |          | <i>LL_OPAMP_SetTrimmingMode</i>               |
|          | VMSEL    | <i>LL_OPAMP_GetFunctionalMode</i>             |
|          |          | <i>LL_OPAMP_GetInputInverting</i>             |
|          |          | <i>LL_OPAMP_SetFunctionalMode</i>             |
|          |          | <i>LL_OPAMP_SetInputInverting</i>             |
|          | VMSSEL   | <i>LL_OPAMP_GetInputInvertingSecondary</i>    |
|          |          | <i>LL_OPAMP_SetInputInvertingSecondary</i>    |
|          | VPSEL    | <i>LL_OPAMP_GetInputNonInverting</i>          |
|          |          | <i>LL_OPAMP_SetInputNonInverting</i>          |
|          | VPSSEL   | <i>LL_OPAMP_GetInputNonInvertingSecondary</i> |
|          |          | <i>LL_OPAMP_SetInputNonInvertingSecondary</i> |

## 82.14 PWR

Table 38: Correspondence between PWR registers and PWR low-layer driver functions

| Register | Field | Function                          |
|----------|-------|-----------------------------------|
| CR       | CSBF  | <i>LL_PWR_ClearFlag_SB</i>        |
|          | CWUF  | <i>LL_PWR_ClearFlag_WU</i>        |
|          | DBP   | <i>LL_PWR_DisableBkUpAccess</i>   |
|          |       | <i>LL_PWR_EnableBkUpAccess</i>    |
|          |       | <i>LL_PWR_IsEnabledBkUpAccess</i> |
|          | ENSD1 | <i>LL_PWR_EnableSDADC</i>         |
|          |       | <i>LL_PWR_IsEnabledSDADC</i>      |
|          | ENSD2 | <i>LL_PWR_EnableSDADC</i>         |
|          |       | <i>LL_PWR_IsEnabledSDADC</i>      |
|          | ENSD3 | <i>LL_PWR_EnableSDADC</i>         |
|          |       | <i>LL_PWR_IsEnabledSDADC</i>      |
|          | LPDS  | <i>LL_PWR_GetPowerMode</i>        |
|          |       | <i>LL_PWR_GetRegulModeDS</i>      |
|          |       | <i>LL_PWR_SetPowerMode</i>        |
|          |       | <i>LL_PWR_SetRegulModeDS</i>      |
|          | PDDS  | <i>LL_PWR_GetPowerMode</i>        |
|          |       | <i>LL_PWR_SetPowerMode</i>        |
|          | PLS   | <i>LL_PWR_GetPVDLevel</i>         |
|          |       | <i>LL_PWR_SetPVDLevel</i>         |

| Register | Field       | Function                              |
|----------|-------------|---------------------------------------|
| CSR      | PVDE        | <i>LL_PWR_DisablePVD</i>              |
|          |             | <i>LL_PWR_EnablePVD</i>               |
|          |             | <i>LL_PWR_IsEnabledPVD</i>            |
|          | EWUP1       | <i>LL_PWR_DisableWakeUpPin</i>        |
|          |             | <i>LL_PWR_EnableWakeUpPin</i>         |
|          |             | <i>LL_PWR_IsEnabledWakeUpPin</i>      |
|          | EWUP2       | <i>LL_PWR_DisableWakeUpPin</i>        |
|          |             | <i>LL_PWR_EnableWakeUpPin</i>         |
|          |             | <i>LL_PWR_IsEnabledWakeUpPin</i>      |
|          | EWUP3       | <i>LL_PWR_DisableWakeUpPin</i>        |
|          |             | <i>LL_PWR_EnableWakeUpPin</i>         |
|          |             | <i>LL_PWR_IsEnabledWakeUpPin</i>      |
|          | PVDO        | <i>LL_PWR_IsActiveFlag_PVDO</i>       |
|          | SBF         | <i>LL_PWR_IsActiveFlag_SB</i>         |
|          | VREFINTRDYF | <i>LL_PWR_IsActiveFlag_VREFINTRDY</i> |
|          | WUF         | <i>LL_PWR_IsActiveFlag_WU</i>         |

## 82.15 RCC

Table 39: Correspondence between RCC registers and RCC low-layer driver functions

| Register | Field  | Function                               |
|----------|--------|----------------------------------------|
| BDCR     | BDRST  | <i>LL_RCC_ForceBackupDomainReset</i>   |
|          |        | <i>LL_RCC_ReleaseBackupDomainReset</i> |
|          | LSEBYP | <i>LL_RCC_LSE_DisableBypass</i>        |
|          |        | <i>LL_RCC_LSE_EnableBypass</i>         |
|          | LSEDRV | <i>LL_RCC_LSE_GetDriveCapability</i>   |
|          |        | <i>LL_RCC_LSE_SetDriveCapability</i>   |
|          | LSEON  | <i>LL_RCC_LSE_Disable</i>              |
|          |        | <i>LL_RCC_LSE_Enable</i>               |
|          | LSERDY | <i>LL_RCC_LSE_IsReady</i>              |
|          | RTCEN  | <i>LL_RCC_DisableRTC</i>               |
|          |        | <i>LL_RCC_EnableRTC</i>                |
|          |        | <i>LL_RCC_IsEnabledRTC</i>             |
|          | RTCSEL | <i>LL_RCC_GetRTCClockSource</i>        |
|          |        | <i>LL_RCC_SetRTCClockSource</i>        |
|          | CFGR   | <i>LL_RCC_GetAHBPrescaler</i>          |
|          |        | <i>LL_RCC_SetAHBPrescaler</i>          |

| Register | Field    | Function                                                           |
|----------|----------|--------------------------------------------------------------------|
| CFGR1    | I2SSRC   | <i>LL_RCC_GetI2SClockSource</i><br><i>LL_RCC_SetI2SClockSource</i> |
|          | MCO      | <i>LL_RCC_ConfigMCO</i>                                            |
|          | MCOF     | <i>LL_RCC_IsActiveFlag_MCO1</i>                                    |
|          | MCOPRE   | <i>LL_RCC_ConfigMCO</i>                                            |
|          | PLLMUL   | <i>LL_RCC_PLL_ConfigDomain_SYS</i>                                 |
|          |          | <i>LL_RCC_PLL_GetMultiplicator</i>                                 |
|          | PLLNODIV | <i>LL_RCC_ConfigMCO</i>                                            |
|          | PLLSRC   | <i>LL_RCC_PLL_ConfigDomain_SYS</i>                                 |
|          |          | <i>LL_RCC_PLL_GetMainSource</i>                                    |
|          | PPRE1    | <i>LL_RCC_GetAPB1Prescaler</i><br><i>LL_RCC_SetAPB1Prescaler</i>   |
|          |          | <i>LL_RCC_GetAPB2Prescaler</i><br><i>LL_RCC_SetAPB2Prescaler</i>   |
|          | SW       | <i>LL_RCC_SetSysClkSource</i>                                      |
|          | SWS      | <i>LL_RCC_GetSysClkSource</i>                                      |
|          | USBPRE   | <i>LL_RCC_GetUSBClockSource</i><br><i>LL_RCC_SetUSBClockSource</i> |
|          |          |                                                                    |
| CFGR2    | ADCPRE12 | <i>LL_RCC_GetADCClockSource</i><br><i>LL_RCC_SetADCClockSource</i> |
|          |          | <i>LL_RCC_GetADCClockSource</i><br><i>LL_RCC_SetADCClockSource</i> |
|          | PREDIV   | <i>LL_RCC_PLL_ConfigDomain_SYS</i>                                 |
|          |          | <i>LL_RCC_PLL_GetPrediv</i>                                        |
| CFGR3    | I2C1SW   | <i>LL_RCC_GetI2CClockSource</i><br><i>LL_RCC_SetI2CClockSource</i> |
|          |          | <i>LL_RCC_GetI2CClockSource</i><br><i>LL_RCC_SetI2CClockSource</i> |
|          | I2C3SW   | <i>LL_RCC_GetI2CClockSource</i><br><i>LL_RCC_SetI2CClockSource</i> |
|          |          | <i>LL_RCC_GetI2CClockSource</i><br><i>LL_RCC_SetI2CClockSource</i> |
|          | TIM15SW  | <i>LL_RCC_GetTIMClockSource</i><br><i>LL_RCC_SetTIMClockSource</i> |
|          |          | <i>LL_RCC_GetTIMClockSource</i><br><i>LL_RCC_SetTIMClockSource</i> |
|          | TIM16SW  | <i>LL_RCC_GetTIMClockSource</i><br><i>LL_RCC_SetTIMClockSource</i> |
|          |          | <i>LL_RCC_GetTIMClockSource</i><br><i>LL_RCC_SetTIMClockSource</i> |
|          | TIM17SW  | <i>LL_RCC_GetTIMClockSource</i><br><i>LL_RCC_SetTIMClockSource</i> |
|          |          |                                                                    |

| Register | Field    | Function                          |
|----------|----------|-----------------------------------|
| CIR      | TIM1SW   | <i>LL_RCC_GetTIMClockSource</i>   |
|          |          | <i>LL_RCC_SetTIMClockSource</i>   |
|          | TIM20SW  | <i>LL_RCC_GetTIMClockSource</i>   |
|          |          | <i>LL_RCC_SetTIMClockSource</i>   |
|          | TIM2SW   | <i>LL_RCC_GetTIMClockSource</i>   |
|          |          | <i>LL_RCC_SetTIMClockSource</i>   |
|          | TIM34SW  | <i>LL_RCC_GetTIMClockSource</i>   |
|          |          | <i>LL_RCC_SetTIMClockSource</i>   |
|          | TIM8SW   | <i>LL_RCC_GetTIMClockSource</i>   |
|          |          | <i>LL_RCC_SetTIMClockSource</i>   |
|          | UART4SW  | <i>LL_RCC_GetUARTClockSource</i>  |
|          |          | <i>LL_RCC_SetUARTClockSource</i>  |
|          | UART5SW  | <i>LL_RCC_GetUARTClockSource</i>  |
|          |          | <i>LL_RCC_SetUARTClockSource</i>  |
|          | USART1SW | <i>LL_RCC_GetUSARTClockSource</i> |
|          |          | <i>LL_RCC_SetUSARTClockSource</i> |
|          | USART2SW | <i>LL_RCC_GetUSARTClockSource</i> |
|          |          | <i>LL_RCC_SetUSARTClockSource</i> |
|          | USART3SW | <i>LL_RCC_GetUSARTClockSource</i> |
|          |          | <i>LL_RCC_SetUSARTClockSource</i> |
|          | CSSC     | <i>LL_RCC_ClearFlag_HSECSS</i>    |
|          | CSSF     | <i>LL_RCC_IsActiveFlag_HSECSS</i> |
|          | HSERDYC  | <i>LL_RCC_ClearFlag_HSERDY</i>    |
|          | HSERDYF  | <i>LL_RCC_IsActiveFlag_HSERDY</i> |
|          | HSERDYIE | <i>LL_RCC_DisableIT_HSERDY</i>    |
|          |          | <i>LL_RCC_EnableIT_HSERDY</i>     |
|          |          | <i>LL_RCC_IsEnabledIT_HSERDY</i>  |
|          | HSIRDYC  | <i>LL_RCC_ClearFlag_HSIRDY</i>    |
|          | HSIRDYF  | <i>LL_RCC_IsActiveFlag_HSIRDY</i> |
|          | HSIRDYIE | <i>LL_RCC_DisableIT_HSIRDY</i>    |
|          |          | <i>LL_RCC_EnableIT_HSIRDY</i>     |
|          |          | <i>LL_RCC_IsEnabledIT_HSIRDY</i>  |
|          | LSERDYC  | <i>LL_RCC_ClearFlag_LSERDY</i>    |
|          | LSERDYF  | <i>LL_RCC_IsActiveFlag_LSERDY</i> |
|          | LSERDYIE | <i>LL_RCC_DisableIT_LSERDY</i>    |
|          |          | <i>LL_RCC_EnableIT_LSERDY</i>     |

| Register | Field    | Function                                                                                                                                 |
|----------|----------|------------------------------------------------------------------------------------------------------------------------------------------|
| CR       | LSIRDYC  | <i>LL_RCC_IsEnabledIT_LSIRDY</i>                                                                                                         |
|          | LSIRDYF  | <i>LL_RCC_ClearFlag_LSIRDY</i>                                                                                                           |
|          | LSIRDYIE | <i>LL_RCC_IsActiveFlag_LSIRDY</i><br><i>LL_RCC_DisableIT_LSIRDY</i><br><i>LL_RCC_EnableIT_LSIRDY</i><br><i>LL_RCC_IsEnabledIT_LSIRDY</i> |
|          | PLLRDYC  | <i>LL_RCC_ClearFlag_PLLRDY</i>                                                                                                           |
|          | PLLRDYF  | <i>LL_RCC_IsActiveFlag_PLLRDY</i>                                                                                                        |
|          | PLLRDYIE | <i>LL_RCC_DisableIT_PLLRDY</i><br><i>LL_RCC_EnableIT_PLLRDY</i><br><i>LL_RCC_IsEnabledIT_PLLRDY</i>                                      |
|          | CSSON    | <i>LL_RCC_HSE_DisableCSS</i><br><i>LL_RCC_HSE_EnableCSS</i>                                                                              |
|          | HSEBYP   | <i>LL_RCC_HSE_DisableBypass</i><br><i>LL_RCC_HSE_EnableBypass</i>                                                                        |
|          | HSEON    | <i>LL_RCC_HSE_Disable</i><br><i>LL_RCC_HSE_Enable</i>                                                                                    |
|          | HSERDY   | <i>LL_RCC_HSE_IsReady</i>                                                                                                                |
|          | HSICAL   | <i>LL_RCC_HSI_GetCalibration</i>                                                                                                         |
|          | HSION    | <i>LL_RCC_HSI_Disable</i><br><i>LL_RCC_HSI_Enable</i>                                                                                    |
|          | HSIRDY   | <i>LL_RCC_HSI_IsReady</i>                                                                                                                |
|          | HSITRIM  | <i>LL_RCC_HSI_GetCalibTrimming</i><br><i>LL_RCC_HSI_SetCalibTrimming</i>                                                                 |
|          | PLLON    | <i>LL_RCC_PLL_Disable</i><br><i>LL_RCC_PLL_Enable</i>                                                                                    |
|          | PLLRDY   | <i>LL_RCC_PLL_IsReady</i>                                                                                                                |
| CSR      | IWDGRSTF | <i>LL_RCC_IsActiveFlag_IWDGRST</i>                                                                                                       |
|          | LPWRRSTF | <i>LL_RCC_IsActiveFlag_LPWRST</i>                                                                                                        |
|          | LSION    | <i>LL_RCC_LSI_Disable</i><br><i>LL_RCC_LSI_Enable</i>                                                                                    |
|          | LSIRDY   | <i>LL_RCC_LSI_IsReady</i>                                                                                                                |
|          | OBLRSTF  | <i>LL_RCC_IsActiveFlag_OBLRST</i>                                                                                                        |
|          | PINRSTF  | <i>LL_RCC_IsActiveFlag_PINRST</i>                                                                                                        |
|          | PORRSTF  | <i>LL_RCC_IsActiveFlag_PORRST</i>                                                                                                        |
|          | RMVF     | <i>LL_RCC_ClearResetFlags</i>                                                                                                            |

| Register | Field      | Function                             |
|----------|------------|--------------------------------------|
|          | SFTRSTF    | <i>LL_RCC_IsActiveFlag_SFTRST</i>    |
|          | V18PWRRSTF | <i>LL_RCC_IsActiveFlag_V18PWRRST</i> |
|          | WWDGRSTF   | <i>LL_RCC_IsActiveFlag_WWDGRST</i>   |

## 82.16 RTC

Table 40: Correspondence between RTC registers and RTC low-layer driver functions

| Register | Field | Function                      |
|----------|-------|-------------------------------|
| ALRMAR   | DT    | <i>LL_RTC_ALMA_GetDay</i>     |
|          |       | <i>LL_RTC_ALMA_SetDay</i>     |
|          | DU    | <i>LL_RTC_ALMA_GetDay</i>     |
|          |       | <i>LL_RTC_ALMA_GetWeekDay</i> |
|          |       | <i>LL_RTC_ALMA_SetDay</i>     |
|          |       | <i>LL_RTC_ALMA_SetWeekDay</i> |
|          | HT    | <i>LL_RTC_ALMA_ConfigTime</i> |
|          |       | <i>LL_RTC_ALMA_GetHour</i>    |
|          |       | <i>LL_RTC_ALMA_GetTime</i>    |
|          |       | <i>LL_RTC_ALMA_SetHour</i>    |
|          | HU    | <i>LL_RTC_ALMA_ConfigTime</i> |
|          |       | <i>LL_RTC_ALMA_GetHour</i>    |
|          |       | <i>LL_RTC_ALMA_GetTime</i>    |
|          |       | <i>LL_RTC_ALMA_SetHour</i>    |
|          | MNT   | <i>LL_RTC_ALMA_ConfigTime</i> |
|          |       | <i>LL_RTC_ALMA_GetMinute</i>  |
|          |       | <i>LL_RTC_ALMA_GetTime</i>    |
|          |       | <i>LL_RTC_ALMA_SetMinute</i>  |
|          | MNU   | <i>LL_RTC_ALMA_ConfigTime</i> |
|          |       | <i>LL_RTC_ALMA_GetMinute</i>  |
|          |       | <i>LL_RTC_ALMA_GetTime</i>    |
|          |       | <i>LL_RTC_ALMA_SetMinute</i>  |
|          | MSK1  | <i>LL_RTC_ALMA_GetMask</i>    |
|          |       | <i>LL_RTC_ALMA_SetMask</i>    |
|          | MSK2  | <i>LL_RTC_ALMA_GetMask</i>    |
|          |       | <i>LL_RTC_ALMA_SetMask</i>    |
|          | MSK3  | <i>LL_RTC_ALMA_GetMask</i>    |
|          |       | <i>LL_RTC_ALMA_SetMask</i>    |
|          | MSK4  | <i>LL_RTC_ALMA_GetMask</i>    |

| Register | Field  | Function                                            |
|----------|--------|-----------------------------------------------------|
| ALRMASSR | PM     | <a href="#"><i>LL_RTC_ALMA_SetMask</i></a>          |
|          |        | <a href="#"><i>LL_RTC_ALMA_ConfigTime</i></a>       |
|          |        | <a href="#"><i>LL_RTC_ALMA_GetTimeFormat</i></a>    |
|          |        | <a href="#"><i>LL_RTC_ALMA_SetTimeFormat</i></a>    |
|          | ST     | <a href="#"><i>LL_RTC_ALMA_ConfigTime</i></a>       |
|          |        | <a href="#"><i>LL_RTC_ALMA_GetSecond</i></a>        |
|          |        | <a href="#"><i>LL_RTC_ALMA_GetTime</i></a>          |
|          |        | <a href="#"><i>LL_RTC_ALMA_SetSecond</i></a>        |
|          | SU     | <a href="#"><i>LL_RTC_ALMA_ConfigTime</i></a>       |
|          |        | <a href="#"><i>LL_RTC_ALMA_GetSecond</i></a>        |
|          |        | <a href="#"><i>LL_RTC_ALMA_GetTime</i></a>          |
|          |        | <a href="#"><i>LL_RTC_ALMA_SetSecond</i></a>        |
|          | WDSEL  | <a href="#"><i>LL_RTC_ALMA_DisableWeekday</i></a>   |
|          |        | <a href="#"><i>LL_RTC_ALMA_EnableWeekday</i></a>    |
| ALRMBR   | MASKSS | <a href="#"><i>LL_RTC_ALMA_GetSubSecondMask</i></a> |
|          |        | <a href="#"><i>LL_RTC_ALMA_SetSubSecondMask</i></a> |
|          | SS     | <a href="#"><i>LL_RTC_ALMA_GetSubSecond</i></a>     |
|          |        | <a href="#"><i>LL_RTC_ALMA_SetSubSecond</i></a>     |
|          | DT     | <a href="#"><i>LL_RTC_ALMB_GetDay</i></a>           |
|          |        | <a href="#"><i>LL_RTC_ALMB_SetDay</i></a>           |
|          | DU     | <a href="#"><i>LL_RTC_ALMB_GetDay</i></a>           |
|          |        | <a href="#"><i>LL_RTC_ALMB_GetWeekDay</i></a>       |
|          |        | <a href="#"><i>LL_RTC_ALMB_SetDay</i></a>           |
|          |        | <a href="#"><i>LL_RTC_ALMB_SetWeekDay</i></a>       |
|          | HT     | <a href="#"><i>LL_RTC_ALMB_ConfigTime</i></a>       |
|          |        | <a href="#"><i>LL_RTC_ALMB_GetHour</i></a>          |
|          |        | <a href="#"><i>LL_RTC_ALMB_GetTime</i></a>          |
|          |        | <a href="#"><i>LL_RTC_ALMB_SetHour</i></a>          |
|          | HU     | <a href="#"><i>LL_RTC_ALMB_ConfigTime</i></a>       |
|          |        | <a href="#"><i>LL_RTC_ALMB_GetHour</i></a>          |
|          |        | <a href="#"><i>LL_RTC_ALMB_GetTime</i></a>          |
|          |        | <a href="#"><i>LL_RTC_ALMB_SetHour</i></a>          |
|          | MNT    | <a href="#"><i>LL_RTC_ALMB_ConfigTime</i></a>       |
|          |        | <a href="#"><i>LL_RTC_ALMB_GetMinute</i></a>        |
|          |        | <a href="#"><i>LL_RTC_ALMB_GetTime</i></a>          |
|          |        | <a href="#"><i>LL_RTC_ALMB_SetMinute</i></a>        |

| Register | Field  | Function                            |
|----------|--------|-------------------------------------|
|          | MNU    | <i>LL_RTC_ALMB_ConfigTime</i>       |
|          |        | <i>LL_RTC_ALMB_GetMinute</i>        |
|          |        | <i>LL_RTC_ALMB_GetTime</i>          |
|          |        | <i>LL_RTC_ALMB_SetMinute</i>        |
|          | MSK1   | <i>LL_RTC_ALMB_GetMask</i>          |
|          |        | <i>LL_RTC_ALMB_SetMask</i>          |
|          | MSK2   | <i>LL_RTC_ALMB_GetMask</i>          |
|          |        | <i>LL_RTC_ALMB_SetMask</i>          |
|          | MSK3   | <i>LL_RTC_ALMB_GetMask</i>          |
|          |        | <i>LL_RTC_ALMB_SetMask</i>          |
|          | MSK4   | <i>LL_RTC_ALMB_GetMask</i>          |
|          |        | <i>LL_RTC_ALMB_SetMask</i>          |
|          | PM     | <i>LL_RTC_ALMB_ConfigTime</i>       |
|          |        | <i>LL_RTC_ALMB_GetTimeFormat</i>    |
|          |        | <i>LL_RTC_ALMB_SetTimeFormat</i>    |
|          | ST     | <i>LL_RTC_ALMB_ConfigTime</i>       |
|          |        | <i>LL_RTC_ALMB_GetSecond</i>        |
|          |        | <i>LL_RTC_ALMB_GetTime</i>          |
|          |        | <i>LL_RTC_ALMB_SetSecond</i>        |
|          | SU     | <i>LL_RTC_ALMB_ConfigTime</i>       |
|          |        | <i>LL_RTC_ALMB_GetSecond</i>        |
|          |        | <i>LL_RTC_ALMB_GetTime</i>          |
|          |        | <i>LL_RTC_ALMB_SetSecond</i>        |
|          | WDSEL  | <i>LL_RTC_ALMB_DisableWeekday</i>   |
|          |        | <i>LL_RTC_ALMB_EnableWeekday</i>    |
| ALRMBSSR | MASKSS | <i>LL_RTC_ALMB_GetSubSecondMask</i> |
|          |        | <i>LL_RTC_ALMB_SetSubSecondMask</i> |
|          | SS     | <i>LL_RTC_ALMB_GetSubSecond</i>     |
|          |        | <i>LL_RTC_ALMB_SetSubSecond</i>     |
| BKPxR    | BKP    | <i>LL_RTC_BAK_GetRegister</i>       |
|          |        | <i>LL_RTC_BAK_SetRegister</i>       |
| CALR     | CALM   | <i>LL_RTC_CAL_GetMinus</i>          |
|          |        | <i>LL_RTC_CAL_SetMinus</i>          |
|          | CALP   | <i>LL_RTC_CAL_IsPulseInserted</i>   |
|          |        | <i>LL_RTC_CAL_SetPulse</i>          |
|          | CALW16 | <i>LL_RTC_CAL_GetPeriod</i>         |

| Register | Field   | Function                                                  |
|----------|---------|-----------------------------------------------------------|
| CR       | CALW8   | <a href="#"><i>LL_RTC_CAL_SetPeriod</i></a>               |
|          |         | <a href="#"><i>LL_RTC_CAL_GetPeriod</i></a>               |
|          |         | <a href="#"><i>LL_RTC_CAL_SetPeriod</i></a>               |
| CR       | ADD1H   | <a href="#"><i>LL_RTC_TIME_IncHour</i></a>                |
|          | ALRAE   | <a href="#"><i>LL_RTC_ALMA_Disable</i></a>                |
|          |         | <a href="#"><i>LL_RTC_ALMA_Enable</i></a>                 |
|          | ALRAIE  | <a href="#"><i>LL_RTC_DisableIT_ALRA</i></a>              |
|          |         | <a href="#"><i>LL_RTC_EnableIT_ALRA</i></a>               |
|          |         | <a href="#"><i>LL_RTC_IsEnabledIT_ALRA</i></a>            |
|          | ALRBE   | <a href="#"><i>LL_RTC_ALMB_Disable</i></a>                |
|          |         | <a href="#"><i>LL_RTC_ALMB_Enable</i></a>                 |
|          | ALRBIE  | <a href="#"><i>LL_RTC_DisableIT_ALRB</i></a>              |
|          |         | <a href="#"><i>LL_RTC_EnableIT_ALRB</i></a>               |
|          |         | <a href="#"><i>LL_RTC_IsEnabledIT_ALRB</i></a>            |
| CR       | BKP     | <a href="#"><i>LL_RTC_TIME_DisableDayLightStore</i></a>   |
|          |         | <a href="#"><i>LL_RTC_TIME_EnableDayLightStore</i></a>    |
|          |         | <a href="#"><i>LL_RTC_TIME_IsDayLightStoreEnabled</i></a> |
|          | BYPSHAD | <a href="#"><i>LL_RTC_DisableShadowRegBypass</i></a>      |
|          |         | <a href="#"><i>LL_RTC_EnableShadowRegBypass</i></a>       |
|          |         | <a href="#"><i>LL_RTC_IsShadowRegBypassEnabled</i></a>    |
|          | COE     | <a href="#"><i>LL_RTC_CAL_GetOutputFreq</i></a>           |
|          |         | <a href="#"><i>LL_RTC_CAL_SetOutputFreq</i></a>           |
|          | COSEL   | <a href="#"><i>LL_RTC_CAL_GetOutputFreq</i></a>           |
|          |         | <a href="#"><i>LL_RTC_CAL_SetOutputFreq</i></a>           |
| CR       | FMT     | <a href="#"><i>LL_RTC_GetHourFormat</i></a>               |
|          |         | <a href="#"><i>LL_RTC_SetHourFormat</i></a>               |
|          | OSEL    | <a href="#"><i>LL_RTC_GetAlarmOutEvent</i></a>            |
|          |         | <a href="#"><i>LL_RTC_SetAlarmOutEvent</i></a>            |
|          | POL     | <a href="#"><i>LL_RTC_GetOutputPolarity</i></a>           |
|          |         | <a href="#"><i>LL_RTC_SetOutputPolarity</i></a>           |
|          | REFCKON | <a href="#"><i>LL_RTC_DisableRefClock</i></a>             |
|          |         | <a href="#"><i>LL_RTC_EnableRefClock</i></a>              |
|          | SUB1H   | <a href="#"><i>LL_RTC_TIME_DecHour</i></a>                |
| CR       | TSE     | <a href="#"><i>LL_RTC_TS_Disable</i></a>                  |
|          |         | <a href="#"><i>LL_RTC_TS_Enable</i></a>                   |
|          | TSEDGE  | <a href="#"><i>LL_RTC_TS_GetActiveEdge</i></a>            |

| Register | Field   | Function                       |
|----------|---------|--------------------------------|
| DR       | TSIE    | <i>LL_RTC_TS_SetActiveEdge</i> |
|          |         | <i>LL_RTC_DisableIT_TS</i>     |
|          |         | <i>LL_RTC_EnableIT_TS</i>      |
|          |         | <i>LL_RTC_IsEnabledIT_TS</i>   |
|          | WUCKSEL | <i>LL_RTC_WAKEUP_GetClock</i>  |
|          |         | <i>LL_RTC_WAKEUP_SetClock</i>  |
|          | WUTE    | <i>LL_RTC_WAKEUP_Disable</i>   |
|          |         | <i>LL_RTC_WAKEUP_Enable</i>    |
|          |         | <i>LL_RTC_WAKEUP_IsEnabled</i> |
|          | WUTIE   | <i>LL_RTC_DisableIT_WUT</i>    |
|          |         | <i>LL_RTC_EnableIT_WUT</i>     |
|          |         | <i>LL_RTC_IsEnabledIT_WUT</i>  |
| DR       | DT      | <i>LL_RTC_DATE_Config</i>      |
|          |         | <i>LL_RTC_DATE_Get</i>         |
|          |         | <i>LL_RTC_DATE_GetDay</i>      |
|          |         | <i>LL_RTC_DATE_SetDay</i>      |
|          | DU      | <i>LL_RTC_DATE_Config</i>      |
|          |         | <i>LL_RTC_DATE_Get</i>         |
|          |         | <i>LL_RTC_DATE_GetDay</i>      |
|          |         | <i>LL_RTC_DATE_SetDay</i>      |
|          | MT      | <i>LL_RTC_DATE_Config</i>      |
|          |         | <i>LL_RTC_DATE_Get</i>         |
|          |         | <i>LL_RTC_DATE_GetMonth</i>    |
|          |         | <i>LL_RTC_DATE_SetMonth</i>    |
|          | MU      | <i>LL_RTC_DATE_Config</i>      |
|          |         | <i>LL_RTC_DATE_Get</i>         |
|          |         | <i>LL_RTC_DATE_GetMonth</i>    |
|          |         | <i>LL_RTC_DATE_SetMonth</i>    |
|          | WDU     | <i>LL_RTC_DATE_Config</i>      |
|          |         | <i>LL_RTC_DATE_Get</i>         |
|          |         | <i>LL_RTC_DATE_GetWeekDay</i>  |
|          |         | <i>LL_RTC_DATE_SetWeekDay</i>  |
|          | YT      | <i>LL_RTC_DATE_Config</i>      |
|          |         | <i>LL_RTC_DATE_Get</i>         |
|          |         | <i>LL_RTC_DATE_GetYear</i>     |
|          |         | <i>LL_RTC_DATE_SetYear</i>     |

| Register | Field    | Function                                          |
|----------|----------|---------------------------------------------------|
| YU       | YU       | <a href="#"><i>LL_RTC_DATE_Config</i></a>         |
|          |          | <a href="#"><i>LL_RTC_DATE_Get</i></a>            |
|          |          | <a href="#"><i>LL_RTC_DATE_GetYear</i></a>        |
|          |          | <a href="#"><i>LL_RTC_DATE_SetYear</i></a>        |
| ISR      | ALRAF    | <a href="#"><i>LL_RTC_ClearFlag_ALRA</i></a>      |
|          |          | <a href="#"><i>LL_RTC_IsActiveFlag_ALRA</i></a>   |
|          | ALRAWF   | <a href="#"><i>LL_RTC_IsActiveFlag_ALRAW</i></a>  |
|          | ALRBF    | <a href="#"><i>LL_RTC_ClearFlag_ALRB</i></a>      |
|          |          | <a href="#"><i>LL_RTC_IsActiveFlag_ALRB</i></a>   |
|          | ALRBWF   | <a href="#"><i>LL_RTC_IsActiveFlag_ALRBW</i></a>  |
|          | INIT     | <a href="#"><i>LL_RTC_DisableInitMode</i></a>     |
|          |          | <a href="#"><i>LL_RTC_EnableInitMode</i></a>      |
|          | INITF    | <a href="#"><i>LL_RTC_IsActiveFlag_INIT</i></a>   |
|          | INITS    | <a href="#"><i>LL_RTC_IsActiveFlag_INITS</i></a>  |
|          | RECALPF  | <a href="#"><i>LL_RTC_IsActiveFlag_RECALP</i></a> |
|          | RSF      | <a href="#"><i>LL_RTC_ClearFlag_RS</i></a>        |
|          |          | <a href="#"><i>LL_RTC_IsActiveFlag_RS</i></a>     |
|          | SHPF     | <a href="#"><i>LL_RTC_IsActiveFlag_SHP</i></a>    |
|          | TAMP1F   | <a href="#"><i>LL_RTC_ClearFlag_TAMP1</i></a>     |
|          |          | <a href="#"><i>LL_RTC_IsActiveFlag_TAMP1</i></a>  |
|          | TAMP2F   | <a href="#"><i>LL_RTC_ClearFlag_TAMP2</i></a>     |
|          |          | <a href="#"><i>LL_RTC_IsActiveFlag_TAMP2</i></a>  |
|          | TAMP3F   | <a href="#"><i>LL_RTC_ClearFlag_TAMP3</i></a>     |
|          |          | <a href="#"><i>LL_RTC_IsActiveFlag_TAMP3</i></a>  |
|          | TSF      | <a href="#"><i>LL_RTC_ClearFlag_TS</i></a>        |
|          |          | <a href="#"><i>LL_RTC_IsActiveFlag_TS</i></a>     |
|          | TSOVF    | <a href="#"><i>LL_RTC_ClearFlag_TSOV</i></a>      |
|          |          | <a href="#"><i>LL_RTC_IsActiveFlag_TSOV</i></a>   |
|          | WUTF     | <a href="#"><i>LL_RTC_ClearFlag_WUT</i></a>       |
|          |          | <a href="#"><i>LL_RTC_IsActiveFlag_WUT</i></a>    |
|          | WUTWF    | <a href="#"><i>LL_RTC_IsActiveFlag_WUTW</i></a>   |
| PRER     | PREDIV_A | <a href="#"><i>LL_RTC_GetAsynchPrescaler</i></a>  |
|          |          | <a href="#"><i>LL_RTC_SetAsynchPrescaler</i></a>  |
|          | PREDIV_S | <a href="#"><i>LL_RTC_GetSynchPrescaler</i></a>   |
|          |          | <a href="#"><i>LL_RTC_SetSynchPrescaler</i></a>   |
| SHIFTR   | ADD1S    | <a href="#"><i>LL_RTC_TIME_Synchronize</i></a>    |

| Register | Field        | Function                                |
|----------|--------------|-----------------------------------------|
|          | SUBFS        | <i>LL_RTC_TIME_Synchronize</i>          |
| SSR      | SS           | <i>LL_RTC_TIME_GetSubSecond</i>         |
| TAFCR    | ALARMOUTTYPE | <i>LL_RTC_GetAlarmOutputType</i>        |
|          |              | <i>LL_RTC_SetAlarmOutputType</i>        |
|          | PC13MODE     | <i>LL_RTC_DisablePushPullMode</i>       |
|          |              | <i>LL_RTC_EnablePushPullMode</i>        |
|          | PC14MODE     | <i>LL_RTC_DisablePushPullMode</i>       |
|          |              | <i>LL_RTC_EnablePushPullMode</i>        |
|          | PC14VALUE    | <i>LL_RTC_ResetOutputPin</i>            |
|          |              | <i>LL_RTC_SetOutputPin</i>              |
|          | PC15MODE     | <i>LL_RTC_DisablePushPullMode</i>       |
|          |              | <i>LL_RTC_EnablePushPullMode</i>        |
|          | PC15VALUE    | <i>LL_RTC_ResetOutputPin</i>            |
|          |              | <i>LL_RTC_SetOutputPin</i>              |
|          | TAMP1E       | <i>LL_RTC_TAMPER_Disable</i>            |
|          |              | <i>LL_RTC_TAMPER_Enable</i>             |
|          | TAMP1TRG     | <i>LL_RTC_TAMPER_DisableActiveLevel</i> |
|          |              | <i>LL_RTC_TAMPER_EnableActiveLevel</i>  |
|          | TAMP2E       | <i>LL_RTC_TAMPER_Disable</i>            |
|          |              | <i>LL_RTC_TAMPER_Enable</i>             |
|          | TAMP2TRG     | <i>LL_RTC_TAMPER_DisableActiveLevel</i> |
|          |              | <i>LL_RTC_TAMPER_EnableActiveLevel</i>  |
|          | TAMP3E       | <i>LL_RTC_TAMPER_Disable</i>            |
|          |              | <i>LL_RTC_TAMPER_Enable</i>             |
|          | TAMP3TRG     | <i>LL_RTC_TAMPER_DisableActiveLevel</i> |
|          |              | <i>LL_RTC_TAMPER_EnableActiveLevel</i>  |
|          | TAMPFLT      | <i>LL_RTC_TAMPER_GetFilterCount</i>     |
|          |              | <i>LL_RTC_TAMPER_SetFilterCount</i>     |
|          | TAMPFREQ     | <i>LL_RTC_TAMPER_GetSamplingFreq</i>    |
|          |              | <i>LL_RTC_TAMPER_SetSamplingFreq</i>    |
|          | TAMPIE       | <i>LL_RTC_DisableIT_TAMP</i>            |
|          |              | <i>LL_RTC_EnableIT_TAMP</i>             |
|          |              | <i>LL_RTC_IsEnabledIT_TAMP</i>          |
|          | TAMPPRCH     | <i>LL_RTC_TAMPER_GetPrecharge</i>       |
|          |              | <i>LL_RTC_TAMPER_SetPrecharge</i>       |
|          | TAMPPUDIS    | <i>LL_RTC_TAMPER_DisablePullUp</i>      |

| Register | Field | Function                                          |
|----------|-------|---------------------------------------------------|
| TAMPTS   |       | <a href="#"><i>LL_RTC_TAMPER_EnablePullUp</i></a> |
|          |       | <a href="#"><i>LL_RTC_TS_DisableOnTamper</i></a>  |
|          |       | <a href="#"><i>LL_RTC_TS_EnableOnTamper</i></a>   |
| TR       | HT    | <a href="#"><i>LL_RTC_TIME_Config</i></a>         |
|          |       | <a href="#"><i>LL_RTC_TIME_Get</i></a>            |
|          |       | <a href="#"><i>LL_RTC_TIME_GetHour</i></a>        |
|          |       | <a href="#"><i>LL_RTC_TIME_SetHour</i></a>        |
| TR       | HU    | <a href="#"><i>LL_RTC_TIME_Config</i></a>         |
|          |       | <a href="#"><i>LL_RTC_TIME_Get</i></a>            |
|          |       | <a href="#"><i>LL_RTC_TIME_GetHour</i></a>        |
|          |       | <a href="#"><i>LL_RTC_TIME_SetHour</i></a>        |
| TR       | MNT   | <a href="#"><i>LL_RTC_TIME_Config</i></a>         |
|          |       | <a href="#"><i>LL_RTC_TIME_Get</i></a>            |
|          |       | <a href="#"><i>LL_RTC_TIME_GetMinute</i></a>      |
|          |       | <a href="#"><i>LL_RTC_TIME_SetMinute</i></a>      |
| TR       | MNU   | <a href="#"><i>LL_RTC_TIME_Config</i></a>         |
|          |       | <a href="#"><i>LL_RTC_TIME_Get</i></a>            |
|          |       | <a href="#"><i>LL_RTC_TIME_GetMinute</i></a>      |
|          |       | <a href="#"><i>LL_RTC_TIME_SetMinute</i></a>      |
| TR       | PM    | <a href="#"><i>LL_RTC_TIME_Config</i></a>         |
|          |       | <a href="#"><i>LL_RTC_TIME_GetFormat</i></a>      |
|          |       | <a href="#"><i>LL_RTC_TIME_SetFormat</i></a>      |
|          |       |                                                   |
| TR       | ST    | <a href="#"><i>LL_RTC_TIME_Config</i></a>         |
|          |       | <a href="#"><i>LL_RTC_TIME_Get</i></a>            |
|          |       | <a href="#"><i>LL_RTC_TIME_GetSecond</i></a>      |
|          |       | <a href="#"><i>LL_RTC_TIME_SetSecond</i></a>      |
| TR       | SU    | <a href="#"><i>LL_RTC_TIME_Config</i></a>         |
|          |       | <a href="#"><i>LL_RTC_TIME_Get</i></a>            |
|          |       | <a href="#"><i>LL_RTC_TIME_GetSecond</i></a>      |
|          |       | <a href="#"><i>LL_RTC_TIME_SetSecond</i></a>      |
| TSDR     | DT    | <a href="#"><i>LL_RTC_TS_GetDate</i></a>          |
|          |       | <a href="#"><i>LL_RTC_TS_GetDay</i></a>           |
|          | DU    | <a href="#"><i>LL_RTC_TS_GetDate</i></a>          |
|          |       | <a href="#"><i>LL_RTC_TS_GetDay</i></a>           |
|          | MT    | <a href="#"><i>LL_RTC_TS_GetDate</i></a>          |
|          |       | <a href="#"><i>LL_RTC_TS_GetMonth</i></a>         |

| Register                                            | Field                                              | Function                                             |
|-----------------------------------------------------|----------------------------------------------------|------------------------------------------------------|
| TSTR                                                | MU                                                 | <a href="#"><i>LL_RTC_TS_GetDate</i></a>             |
|                                                     |                                                    | <a href="#"><i>LL_RTC_TS_GetMonth</i></a>            |
|                                                     | WDU                                                | <a href="#"><i>LL_RTC_TS_GetDate</i></a>             |
|                                                     |                                                    | <a href="#"><i>LL_RTC_TS_GetWeekDay</i></a>          |
| TSSSR                                               | SS                                                 | <a href="#"><i>LL_RTC_TS_GetSubSecond</i></a>        |
| HT                                                  | <a href="#"><i>LL_RTC_TS_GetHour</i></a>           |                                                      |
|                                                     | <a href="#"><i>LL_RTC_TS_GetTime</i></a>           |                                                      |
| HU                                                  | <a href="#"><i>LL_RTC_TS_GetHour</i></a>           |                                                      |
|                                                     | <a href="#"><i>LL_RTC_TS_GetTime</i></a>           |                                                      |
| MNT                                                 | <a href="#"><i>LL_RTC_TS_GetMinute</i></a>         |                                                      |
|                                                     | <a href="#"><i>LL_RTC_TS_GetTime</i></a>           |                                                      |
| MNU                                                 | <a href="#"><i>LL_RTC_TS_GetMinute</i></a>         |                                                      |
|                                                     | <a href="#"><i>LL_RTC_TS_GetTime</i></a>           |                                                      |
| PM                                                  | <a href="#"><i>LL_RTC_TS_GetTimeFormat</i></a>     |                                                      |
| ST                                                  | <a href="#"><i>LL_RTC_TS_GetSecond</i></a>         |                                                      |
|                                                     | <a href="#"><i>LL_RTC_TS_GetTime</i></a>           |                                                      |
| SU                                                  | <a href="#"><i>LL_RTC_TS_GetSecond</i></a>         |                                                      |
|                                                     | <a href="#"><i>LL_RTC_TS_GetTime</i></a>           |                                                      |
| WPR                                                 | KEY                                                | <a href="#"><i>LL_RTC_DisableWriteProtection</i></a> |
| <a href="#"><i>LL_RTC_EnableWriteProtection</i></a> |                                                    |                                                      |
| WUT                                                 | <a href="#"><i>LL_RTC_WAKEUP_SetAutoReload</i></a> |                                                      |
|                                                     | <a href="#"><i>LL_RTC_WAKEUP_SetAutoReload</i></a> |                                                      |

## 82.17 SPI

Table 41: Correspondence between SPI registers and SPI low-layer driver functions

| Register | Field    | Function                                           |
|----------|----------|----------------------------------------------------|
| CR1      | BIDIMODE | <a href="#"><i>LL_SPI_SetTransferDirection</i></a> |
|          |          | <a href="#"><i>LL_SPI_SetTransferDirection</i></a> |
|          | BIDIOE   | <a href="#"><i>LL_SPI_SetTransferDirection</i></a> |
|          |          | <a href="#"><i>LL_SPI_SetTransferDirection</i></a> |
|          | BR       | <a href="#"><i>LL_SPI_SetBaudRatePrescaler</i></a> |
|          |          | <a href="#"><i>LL_SPI_SetBaudRatePrescaler</i></a> |
|          | CPHA     | <a href="#"><i>LL_SPI_SetClockPhase</i></a>        |
|          |          | <a href="#"><i>LL_SPI_SetClockPhase</i></a>        |
|          | CPOL     | <a href="#"><i>LL_SPI_SetClockPolarity</i></a>     |
|          |          | <a href="#"><i>LL_SPI_SetClockPolarity</i></a>     |

| Register | Field    | Function                                           |
|----------|----------|----------------------------------------------------|
| CR1      | CRCEN    | <a href="#"><i>LL_SPI_DisableCRC</i></a>           |
|          |          | <a href="#"><i>LL_SPI_EnableCRC</i></a>            |
|          |          | <a href="#"><i>LL_SPI_IsEnabledCRC</i></a>         |
|          | CRCL     | <a href="#"><i>LL_SPI_GetCRCWidth</i></a>          |
|          |          | <a href="#"><i>LL_SPI_SetCRCWidth</i></a>          |
|          | CRCNEXT  | <a href="#"><i>LL_SPI_SetCRCNext</i></a>           |
|          | LSBFIRST | <a href="#"><i>LL_SPI_GetTransferBitOrder</i></a>  |
|          |          | <a href="#"><i>LL_SPI_SetTransferBitOrder</i></a>  |
|          | MSTR     | <a href="#"><i>LL_SPI_GetMode</i></a>              |
|          |          | <a href="#"><i>LL_SPI_SetMode</i></a>              |
| CR2      | RXONLY   | <a href="#"><i>LL_SPI_GetTransferDirection</i></a> |
|          |          | <a href="#"><i>LL_SPI_SetTransferDirection</i></a> |
|          | SPE      | <a href="#"><i>LL_SPI_Disable</i></a>              |
|          |          | <a href="#"><i>LL_SPI_Enable</i></a>               |
|          |          | <a href="#"><i>LL_SPI_IsEnabled</i></a>            |
|          | SSI      | <a href="#"><i>LL_SPI_GetMode</i></a>              |
|          |          | <a href="#"><i>LL_SPI_SetMode</i></a>              |
|          | SSM      | <a href="#"><i>LL_SPI_GetNSSMode</i></a>           |
|          |          | <a href="#"><i>LL_SPI_SetNSSMode</i></a>           |
|          | DS       | <a href="#"><i>LL_SPI_GetDataWidth</i></a>         |
|          |          | <a href="#"><i>LL_SPI_SetDataWidth</i></a>         |
| CR3      | ERRIE    | <a href="#"><i>LL_SPI_DisableIT_ERR</i></a>        |
|          |          | <a href="#"><i>LL_SPI_EnableIT_ERR</i></a>         |
|          |          | <a href="#"><i>LL_SPI_IsEnabledIT_ERR</i></a>      |
|          | FRF      | <a href="#"><i>LL_SPI_GetStandard</i></a>          |
|          |          | <a href="#"><i>LL_SPI_SetStandard</i></a>          |
|          | FRXTH    | <a href="#"><i>LL_SPI_GetRxFIFOThreshold</i></a>   |
|          |          | <a href="#"><i>LL_SPI_SetRxFIFOThreshold</i></a>   |
|          | LDMARX   | <a href="#"><i>LL_SPI_GetDMAParity_RX</i></a>      |
|          |          | <a href="#"><i>LL_SPI_SetDMAParity_RX</i></a>      |
|          | LDMATX   | <a href="#"><i>LL_SPI_GetDMAParity_TX</i></a>      |
|          |          | <a href="#"><i>LL_SPI_SetDMAParity_TX</i></a>      |
| CR4      | NSSP     | <a href="#"><i>LL_SPI_DisableNSSPulseMgt</i></a>   |
|          |          | <a href="#"><i>LL_SPI_EnableNSSPulseMgt</i></a>    |
|          |          | <a href="#"><i>LL_SPI_IsEnabledNSSPulse</i></a>    |
|          | RXDMAEN  | <a href="#"><i>LL_SPI_DisableDMAReq_RX</i></a>     |

| Register | Field   | Function                          |
|----------|---------|-----------------------------------|
|          | RXNEIE  | <i>LL_SPI_EnableDMAReq_RX</i>     |
|          |         | <i>LL_SPI_IsEnabledDMAReq_RX</i>  |
|          |         | <i>LL_SPI_DisableIT_RXNE</i>      |
|          | SSOE    | <i>LL_SPI_EnableIT_RXNE</i>       |
|          |         | <i>LL_SPI_IsEnabledIT_RXNE</i>    |
|          |         | <i>LL_SPI_GetNSSMode</i>          |
|          | TXDMAEN | <i>LL_SPI_SetNSSMode</i>          |
|          |         | <i>LL_SPI_DisableDMAReq_TX</i>    |
|          |         | <i>LL_SPI_EnableDMAReq_TX</i>     |
|          | TXEIE   | <i>LL_SPI_IsEnabledDMAReq_TX</i>  |
|          |         | <i>LL_SPI_DisableIT_TXE</i>       |
|          |         | <i>LL_SPI_EnableIT_TXE</i>        |
|          | CRCPR   | <i>LL_SPI_IsEnabledIT_TXE</i>     |
|          |         | <i>LL_SPI_GetCRCPolynomial</i>    |
|          |         | <i>LL_SPI_SetCRCPolynomial</i>    |
| DR       | DR      | <i>LL_SPI_DMA_GetRegAddr</i>      |
|          |         | <i>LL_SPI_ReceiveData16</i>       |
|          |         | <i>LL_SPI_ReceiveData8</i>        |
|          |         | <i>LL_SPI_TransmitData16</i>      |
|          |         | <i>LL_SPI_TransmitData8</i>       |
| RXCRCR   | RXCRC   | <i>LL_SPI_GetRxCRC</i>            |
| SR       | BSY     | <i>LL_SPI_IsActiveFlag_BSY</i>    |
|          | CRCERR  | <i>LL_SPI_ClearFlag_CRCERR</i>    |
|          |         | <i>LL_SPI_IsActiveFlag_CRCERR</i> |
|          | FRE     | <i>LL_SPI_ClearFlag_FRE</i>       |
|          |         | <i>LL_SPI_IsActiveFlag_FRE</i>    |
|          | FRLVL   | <i>LL_SPI_GetRxFIFOLevel</i>      |
|          | FTLVL   | <i>LL_SPI_GetTxFIFOLevel</i>      |
|          | MODF    | <i>LL_SPI_ClearFlag_MODF</i>      |
|          |         | <i>LL_SPI_IsActiveFlag_MODF</i>   |
|          | OVR     | <i>LL_SPI_ClearFlag_OVR</i>       |
|          |         | <i>LL_SPI_IsActiveFlag_OVR</i>    |
|          | RXNE    | <i>LL_SPI_IsActiveFlag_RXNE</i>   |
|          | TXE     | <i>LL_SPI_IsActiveFlag_TXE</i>    |
| TXCRCR   | TXCRC   | <i>LL_SPI_GetTxCRC</i>            |

## 82.18 SYSTEM

Table 42: Correspondence between SYSTEM registers and SYSTEM low-layer driver functions

| Register | Field                  | Function                                           |
|----------|------------------------|----------------------------------------------------|
| APB1_FZ  | DBG_CAN_STOP           | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_I2C1_SMBUS_TIMEOUT | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_I2C2_SMBUS_TIMEOUT | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_I2C3_SMBUS_TIMEOUT | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_IWDG_STOP          | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_RTC_STOP           | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_TIM12_STOP         | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_TIM13_STOP         | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_TIM14_STOP         | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_TIM18_STOP         | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_TIM2_STOP          | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_TIM3_STOP          | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|          | DBG_TIM4_STOP          | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|          |                        | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |

| Register  | Field           | Function                                           |
|-----------|-----------------|----------------------------------------------------|
| APB2_FZ   | DBG_TIM5_STOP   | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|           | DBG_TIM6_STOP   | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|           | DBG_TIM7_STOP   | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|           | DBG_WWDG_STOP   | <a href="#">LL_DBGMCU_APB1_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB1_GRP1_UnFreezePeriph</a> |
|           | DBG_HRTIM1_STOP | <a href="#">LL_DBGMCU_APB2_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB2_GRP1_UnFreezePeriph</a> |
| DBGMCU_CR | DBG_TIM15_STOP  | <a href="#">LL_DBGMCU_APB2_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB2_GRP1_UnFreezePeriph</a> |
|           | DBG_TIM16_STOP  | <a href="#">LL_DBGMCU_APB2_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB2_GRP1_UnFreezePeriph</a> |
|           | DBG_TIM17_STOP  | <a href="#">LL_DBGMCU_APB2_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB2_GRP1_UnFreezePeriph</a> |
|           | DBG_TIM19_STOP  | <a href="#">LL_DBGMCU_APB2_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB2_GRP1_UnFreezePeriph</a> |
|           | DBG_TIM1_STOP   | <a href="#">LL_DBGMCU_APB2_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB2_GRP1_UnFreezePeriph</a> |
|           | DBG_TIM20_STOP  | <a href="#">LL_DBGMCU_APB2_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB2_GRP1_UnFreezePeriph</a> |
|           | DBG_TIM8_STOP   | <a href="#">LL_DBGMCU_APB2_GRP1_FreezePeriph</a>   |
|           |                 | <a href="#">LL_DBGMCU_APB2_GRP1_UnFreezePeriph</a> |
|           | DBG_SLEEP       | <a href="#">LL_DBGMCU_DisableDBGSleepMode</a>      |
|           |                 | <a href="#">LL_DBGMCU_EnableDBGSleepMode</a>       |
|           | DBG_STANDBY     | <a href="#">LL_DBGMCU_DisableDBGStandbyMode</a>    |
|           |                 | <a href="#">LL_DBGMCU_EnableDBGStandbyMode</a>     |
|           | DBG_STOP        | <a href="#">LL_DBGMCU_DisableDBGStopMode</a>       |

| Register       | Field           | Function                                          |
|----------------|-----------------|---------------------------------------------------|
| DBGMCU_ID_CODE | TRACE_IOEN      | <a href="#">LL_DBGMCU_EnableDBGStopMode</a>       |
|                |                 | <a href="#">LL_DBGMCU_GetTracePinAssignment</a>   |
|                |                 | <a href="#">LL_DBGMCU_SetTracePinAssignment</a>   |
|                | TRACE_MODE      | <a href="#">LL_DBGMCU_GetTracePinAssignment</a>   |
|                |                 | <a href="#">LL_DBGMCU_SetTracePinAssignment</a>   |
|                | DEV_ID          | <a href="#">LL_DBGMCU_GetDeviceID</a>             |
|                | REV_ID          | <a href="#">LL_DBGMCU_GetRevisionID</a>           |
| FLASH_ACR      | HLFCYA          | <a href="#">LL_FLASH_DisableHalfCycleAccess</a>   |
|                |                 | <a href="#">LL_FLASH_EnableHalfCycleAccess</a>    |
|                |                 | <a href="#">LL_FLASH_IsHalfCycleAccessEnabled</a> |
|                | LATENCY         | <a href="#">LL_FLASH_GetLatency</a>               |
|                |                 | <a href="#">LL_FLASH_SetLatency</a>               |
|                | PRFTBE          | <a href="#">LL_FLASH_DisablePrefetch</a>          |
|                |                 | <a href="#">LL_FLASH_EnablePrefetch</a>           |
|                | PRFTBS          | <a href="#">LL_FLASH_IsPrefetchEnabled</a>        |
| SYSCFG_CFGR1   | ADC24_DMA_RMP   | <a href="#">LL_SYSCFG_SetRemapDMA_ADC</a>         |
|                | DAC1_TRIG1_RMP  | <a href="#">LL_SYSCFG_SetRemapTrigger_DAC</a>     |
|                | DAC2Ch1_DMA_RMP | <a href="#">LL_SYSCFG_SetRemapDMA_DAC</a>         |
|                | ENCODER_MODE    | <a href="#">LL_SYSCFG_SetRemapInput_TIM</a>       |
|                | FPU_IE_0        | <a href="#">LL_SYSCFG_DisableIT_FPU_IOC</a>       |
|                |                 | <a href="#">LL_SYSCFG_EnableIT_FPU_IOC</a>        |
|                |                 | <a href="#">LL_SYSCFG_IsEnabledIT_FPU_IOC</a>     |
|                | FPU_IE_1        | <a href="#">LL_SYSCFG_DisableIT_FPU_DZC</a>       |
|                |                 | <a href="#">LL_SYSCFG_EnableIT_FPU_DZC</a>        |
|                |                 | <a href="#">LL_SYSCFG_IsEnabledIT_FPU_DZC</a>     |
|                | FPU_IE_2        | <a href="#">LL_SYSCFG_DisableIT_FPU_UFC</a>       |
|                |                 | <a href="#">LL_SYSCFG_EnableIT_FPU_UFC</a>        |
|                |                 | <a href="#">LL_SYSCFG_IsEnabledIT_FPU_UFC</a>     |
|                | FPU_IE_3        | <a href="#">LL_SYSCFG_DisableIT_FPU_OFC</a>       |
|                |                 | <a href="#">LL_SYSCFG_EnableIT_FPU_OFC</a>        |
|                |                 | <a href="#">LL_SYSCFG_IsEnabledIT_FPU_OFC</a>     |
|                | FPU_IE_4        | <a href="#">LL_SYSCFG_DisableIT_FPU_IDC</a>       |
|                |                 | <a href="#">LL_SYSCFG_EnableIT_FPU_IDC</a>        |
|                |                 | <a href="#">LL_SYSCFG_IsEnabledIT_FPU_IDC</a>     |
|                | FPU_IE_5        | <a href="#">LL_SYSCFG_DisableIT_FPU_IJC</a>       |
|                |                 | <a href="#">LL_SYSCFG_EnableIT_FPU_IJC</a>        |

| Register     | Field                | Function                                                                                                                                       |
|--------------|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
|              | I2C1_FMP             | <a href="#">LL_SYSCFG_IsEnabledIT_FPU_IXC</a><br><a href="#">LL_SYSCFG_DisableFastModePlus</a><br><a href="#">LL_SYSCFG_EnableFastModePlus</a> |
|              | I2C2_FMP             | <a href="#">LL_SYSCFG_DisableFastModePlus</a><br><a href="#">LL_SYSCFG_EnableFastModePlus</a>                                                  |
|              | I2C3_FMP             | <a href="#">LL_SYSCFG_DisableFastModePlus</a><br><a href="#">LL_SYSCFG_EnableFastModePlus</a>                                                  |
|              | I2C_PB6_FMP          | <a href="#">LL_SYSCFG_DisableFastModePlus</a><br><a href="#">LL_SYSCFG_EnableFastModePlus</a>                                                  |
|              | I2C_PB7_FMP          | <a href="#">LL_SYSCFG_DisableFastModePlus</a><br><a href="#">LL_SYSCFG_EnableFastModePlus</a>                                                  |
|              | I2C_PB8_FMP          | <a href="#">LL_SYSCFG_DisableFastModePlus</a><br><a href="#">LL_SYSCFG_EnableFastModePlus</a>                                                  |
|              | I2C_PB9_FMP          | <a href="#">LL_SYSCFG_DisableFastModePlus</a><br><a href="#">LL_SYSCFG_EnableFastModePlus</a>                                                  |
|              | MEM_MODE             | <a href="#">LL_SYSCFG_GetRemapMemory</a><br><a href="#">LL_SYSCFG_SetRemapMemory</a>                                                           |
|              | TIM16_DMA_RMP        | <a href="#">LL_SYSCFG_SetRemapDMA_TIM</a>                                                                                                      |
|              | TIM17_DMA_RMP        | <a href="#">LL_SYSCFG_SetRemapDMA_TIM</a>                                                                                                      |
|              | TIM18DAC2Ch1_DMA_RMP | <a href="#">LL_SYSCFG_SetRemapDMA_TIM</a>                                                                                                      |
|              | TIM1_ITR3_RMP        | <a href="#">LL_SYSCFG_SetRemapInput_TIM</a>                                                                                                    |
|              | TIM6DAC1Ch1_DMA_RMP  | <a href="#">LL_SYSCFG_SetRemapDMA_DAC</a><br><a href="#">LL_SYSCFG_SetRemapDMA_TIM</a>                                                         |
|              | TIM7DAC1Ch2_DMA_RMP  | <a href="#">LL_SYSCFG_SetRemapDMA_TIM</a>                                                                                                      |
|              | USB_IT_RMP           | <a href="#">LL_SYSCFG_DisableRemapIT_USB</a><br><a href="#">LL_SYSCFG_EnableRemapIT_USB</a>                                                    |
| SYSCFG_CFGR2 | BYP_ADDR_PAR         | <a href="#">LL_SYSCFG_DisableSRAMParityCheck</a>                                                                                               |
|              | LOCKUP_LOCK          | <a href="#">LL_SYSCFG_GetTIMBreakInputs</a>                                                                                                    |
|              |                      | <a href="#">LL_SYSCFG_SetTIMBreakInputs</a>                                                                                                    |
|              | PVD_LOCK             | <a href="#">LL_SYSCFG_GetTIMBreakInputs</a>                                                                                                    |
|              |                      | <a href="#">LL_SYSCFG_SetTIMBreakInputs</a>                                                                                                    |
|              | SRAM_PARITY_LOCK     | <a href="#">LL_SYSCFG_GetTIMBreakInputs</a>                                                                                                    |
|              |                      | <a href="#">LL_SYSCFG_SetTIMBreakInputs</a>                                                                                                    |
| SYSCFG_CFGR3 | SRAM_PE              | <a href="#">LL_SYSCFG_ClearFlag_SP</a>                                                                                                         |
|              |                      | <a href="#">LL_SYSCFG_IsActiveFlag_SP</a>                                                                                                      |
| SYSCFG_CFGR3 | ADC2_DMA_RMP         | <a href="#">LL_SYSCFG_SetRemapDMA_ADC</a>                                                                                                      |

| Register           | Field         | Function                                      |
|--------------------|---------------|-----------------------------------------------|
|                    | DAC1_TRG3_RMP | <a href="#">LL_SYSCFG_SetRemapTrigger_DAC</a> |
|                    | DAC1_TRG5_RMP | <a href="#">LL_SYSCFG_SetRemapTrigger_DAC</a> |
| SYSCFG_EXTICR<br>1 | EXTI0         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI1         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI10        | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI11        | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI12        | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI13        | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI14        | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI15        | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI2         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI3         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI4         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI5         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI6         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI7         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI8         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
|                    | EXTI9         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |
| SYSCFG_EXTICR<br>2 | EXTI0         | <a href="#">LL_SYSCFG_GetEXTISource</a>       |
|                    |               | <a href="#">LL_SYSCFG_SetEXTISource</a>       |

| Register           | Field  | Function                                                                           |
|--------------------|--------|------------------------------------------------------------------------------------|
|                    | EXTI1  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI10 | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI11 | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI12 | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI13 | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI14 | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI15 | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI2  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI3  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI4  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI5  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI6  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI7  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI8  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI9  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
| SYSCFG_EXTICR<br>3 | EXTI0  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI1  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|                    | EXTI10 | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |

| Register           | Field  | Function                                             |
|--------------------|--------|------------------------------------------------------|
| SYSCFG_EXTICR<br>4 | EXTI11 | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI12 | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI13 | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI14 | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI15 | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI2  | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI3  | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI4  | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI5  | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI6  | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI7  | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI8  | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI9  | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI0  | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI1  | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI10 | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI11 | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |
|                    | EXTI12 | <a href="#"><code>LL_SYSCFG_GetEXTISource</code></a> |
|                    |        | <a href="#"><code>LL_SYSCFG_SetEXTISource</code></a> |

| Register | Field  | Function                                                                           |
|----------|--------|------------------------------------------------------------------------------------|
|          | EXTI13 | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | EXTI14 | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | EXTI15 | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | EXTI2  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | EXTI3  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | EXTI4  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | EXTI5  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | EXTI6  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | EXTI7  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | EXTI8  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | EXTI9  | <a href="#">LL_SYSCFG_GetEXTISource</a><br><a href="#">LL_SYSCFG_SetEXTISource</a> |
|          | PAGE0  | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE1  | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE10 | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE11 | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE12 | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE13 | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE14 | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE15 | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE2  | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE3  | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE4  | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE5  | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE6  | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |
|          | PAGE7  | <a href="#">LL_SYSCFG_EnableCCM_SRAMPageWRP</a>                                    |

| Register | Field | Function                                               |
|----------|-------|--------------------------------------------------------|
|          | PAGE8 | <a href="#"><i>LL_SYSCFG_EnableCCM_SRAMPageWRP</i></a> |
|          | PAGE9 | <a href="#"><i>LL_SYSCFG_EnableCCM_SRAMPageWRP</i></a> |

## 82.19 TIM

Table 43: Correspondence between TIM registers and TIM low-layer driver functions

| Register | Field | Function                                               |
|----------|-------|--------------------------------------------------------|
| ARR      | ARR   | <a href="#"><i>LL_TIM_GetAutoReload</i></a>            |
|          |       | <a href="#"><i>LL_TIM_SetAutoReload</i></a>            |
|          | AOE   | <a href="#"><i>LL_TIM_DisableAutomaticOutput</i></a>   |
|          |       | <a href="#"><i>LL_TIM_EnableAutomaticOutput</i></a>    |
|          |       | <a href="#"><i>LL_TIM_IsEnabledAutomaticOutput</i></a> |
|          | BK2E  | <a href="#"><i>LL_TIM_DisableBRK2</i></a>              |
|          |       | <a href="#"><i>LL_TIM_EnableBRK2</i></a>               |
|          | BK2F  | <a href="#"><i>LL_TIM_ConfigBRK2</i></a>               |
|          | BK2P  | <a href="#"><i>LL_TIM_ConfigBRK2</i></a>               |
|          | BKE   | <a href="#"><i>LL_TIM_DisableBRK</i></a>               |
|          |       | <a href="#"><i>LL_TIM_EnableBRK</i></a>                |
| BDTR     | BKF   | <a href="#"><i>LL_TIM_ConfigBRK</i></a>                |
|          | BKP   | <a href="#"><i>LL_TIM_ConfigBRK</i></a>                |
|          | DTG   | <a href="#"><i>LL_TIM_OC_SetDeadTime</i></a>           |
|          | LOCK  | <a href="#"><i>LL_TIM_CC_SetLockLevel</i></a>          |
|          | MOE   | <a href="#"><i>LL_TIM_DisableAllOutputs</i></a>        |
|          |       | <a href="#"><i>LL_TIM_EnableAllOutputs</i></a>         |
|          |       | <a href="#"><i>LL_TIM_IsEnabledAllOutputs</i></a>      |
|          | OSSI  | <a href="#"><i>LL_TIM_SetOffStates</i></a>             |
|          | OSSR  | <a href="#"><i>LL_TIM_SetOffStates</i></a>             |
| CCER     | CC1E  | <a href="#"><i>LL_TIM_CC_DisableChannel</i></a>        |
|          |       | <a href="#"><i>LL_TIM_CC_EnableChannel</i></a>         |
|          |       | <a href="#"><i>LL_TIM_CC_IsEnabledChannel</i></a>      |
|          | CC1NE | <a href="#"><i>LL_TIM_CC_DisableChannel</i></a>        |
|          |       | <a href="#"><i>LL_TIM_CC_EnableChannel</i></a>         |
|          |       | <a href="#"><i>LL_TIM_CC_IsEnabledChannel</i></a>      |
|          | CC1NP | <a href="#"><i>LL_TIM_IC_Config</i></a>                |
|          |       | <a href="#"><i>LL_TIM_IC_GetPolarity</i></a>           |
|          |       | <a href="#"><i>LL_TIM_IC_SetPolarity</i></a>           |
|          |       | <a href="#"><i>LL_TIM_OC_GetPolarity</i></a>           |

| Register | Field | Function                          |
|----------|-------|-----------------------------------|
| CC1P     |       | <i>LL_TIM_OC_SetPolarity</i>      |
|          |       | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetPolarity</i>      |
|          |       | <i>LL_TIM_IC_SetPolarity</i>      |
|          |       | <i>LL_TIM_OC_ConfigOutput</i>     |
|          |       | <i>LL_TIM_OC_GetPolarity</i>      |
|          |       | <i>LL_TIM_OC_SetPolarity</i>      |
| CC2E     |       | <i>LL_TIM_CC_DisableChannel</i>   |
|          |       | <i>LL_TIM_CC_EnableChannel</i>    |
|          |       | <i>LL_TIM_CC_IsEnabledChannel</i> |
| CC2NE    |       | <i>LL_TIM_CC_DisableChannel</i>   |
|          |       | <i>LL_TIM_CC_EnableChannel</i>    |
|          |       | <i>LL_TIM_CC_IsEnabledChannel</i> |
| CC2NP    |       | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetPolarity</i>      |
|          |       | <i>LL_TIM_IC_SetPolarity</i>      |
|          |       | <i>LL_TIM_OC_GetPolarity</i>      |
|          |       | <i>LL_TIM_OC_SetPolarity</i>      |
| CC2P     |       | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetPolarity</i>      |
|          |       | <i>LL_TIM_IC_SetPolarity</i>      |
|          |       | <i>LL_TIM_OC_ConfigOutput</i>     |
|          |       | <i>LL_TIM_OC_GetPolarity</i>      |
|          |       | <i>LL_TIM_OC_SetPolarity</i>      |
| CC3E     |       | <i>LL_TIM_CC_DisableChannel</i>   |
|          |       | <i>LL_TIM_CC_EnableChannel</i>    |
|          |       | <i>LL_TIM_CC_IsEnabledChannel</i> |
| CC3NE    |       | <i>LL_TIM_CC_DisableChannel</i>   |
|          |       | <i>LL_TIM_CC_EnableChannel</i>    |
|          |       | <i>LL_TIM_CC_IsEnabledChannel</i> |
| CC3NP    |       | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetPolarity</i>      |
|          |       | <i>LL_TIM_IC_SetPolarity</i>      |
|          |       | <i>LL_TIM_OC_GetPolarity</i>      |
|          |       | <i>LL_TIM_OC_SetPolarity</i>      |
| CC3P     |       | <i>LL_TIM_IC_Config</i>           |

| Register | Field | Function                          |
|----------|-------|-----------------------------------|
| CC4E     |       | <i>LL_TIM_IC_GetPolarity</i>      |
|          |       | <i>LL_TIM_IC_SetPolarity</i>      |
|          |       | <i>LL_TIM_OC_ConfigOutput</i>     |
|          |       | <i>LL_TIM_OC_GetPolarity</i>      |
|          |       | <i>LL_TIM_OC_SetPolarity</i>      |
|          |       | <i>LL_TIM_CC_DisableChannel</i>   |
|          |       | <i>LL_TIM_CC_EnableChannel</i>    |
|          |       | <i>LL_TIM_CC_IsEnabledChannel</i> |
|          |       | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetPolarity</i>      |
|          |       | <i>LL_TIM_IC_SetPolarity</i>      |
| CC4P     |       | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetPolarity</i>      |
|          |       | <i>LL_TIM_IC_SetPolarity</i>      |
|          |       | <i>LL_TIM_OC_ConfigOutput</i>     |
|          |       | <i>LL_TIM_OC_GetPolarity</i>      |
|          |       | <i>LL_TIM_OC_SetPolarity</i>      |
| CC5E     |       | <i>LL_TIM_CC_DisableChannel</i>   |
|          |       | <i>LL_TIM_CC_EnableChannel</i>    |
|          |       | <i>LL_TIM_CC_IsEnabledChannel</i> |
| CC5P     |       | <i>LL_TIM_OC_ConfigOutput</i>     |
|          |       | <i>LL_TIM_OC_GetPolarity</i>      |
|          |       | <i>LL_TIM_OC_SetPolarity</i>      |
| CC6E     |       | <i>LL_TIM_CC_DisableChannel</i>   |
|          |       | <i>LL_TIM_CC_EnableChannel</i>    |
|          |       | <i>LL_TIM_CC_IsEnabledChannel</i> |
| CC6P     |       | <i>LL_TIM_OC_ConfigOutput</i>     |
|          |       | <i>LL_TIM_OC_GetPolarity</i>      |
|          |       | <i>LL_TIM_OC_SetPolarity</i>      |
| CCMR1    | CC1S  | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetActiveInput</i>   |
|          |       | <i>LL_TIM_IC_SetActiveInput</i>   |
|          |       | <i>LL_TIM_OC_ConfigOutput</i>     |
|          | CC2S  | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetActiveInput</i>   |
|          |       | <i>LL_TIM_IC_SetActiveInput</i>   |

| Register | Field | Function                          |
|----------|-------|-----------------------------------|
| IC1F     |       | <i>LL_TIM_OC_ConfigOutput</i>     |
|          |       | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetFilter</i>        |
|          |       | <i>LL_TIM_IC_SetFilter</i>        |
| IC1PSC   |       | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetPrescaler</i>     |
|          |       | <i>LL_TIM_IC_SetPrescaler</i>     |
| IC2F     |       | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetFilter</i>        |
|          |       | <i>LL_TIM_IC_SetFilter</i>        |
| IC2PSC   |       | <i>LL_TIM_IC_Config</i>           |
|          |       | <i>LL_TIM_IC_GetPrescaler</i>     |
|          |       | <i>LL_TIM_IC_SetPrescaler</i>     |
| OC1CE    |       | <i>LL_TIM_OC_DisableClear</i>     |
|          |       | <i>LL_TIM_OC_EnableClear</i>      |
|          |       | <i>LL_TIM_OC_IsEnabledClear</i>   |
| OC1FE    |       | <i>LL_TIM_OC_DisableFast</i>      |
|          |       | <i>LL_TIM_OC_EnableFast</i>       |
|          |       | <i>LL_TIM_OC_IsEnabledFast</i>    |
| OC1M     |       | <i>LL_TIM_OC_GetMode</i>          |
|          |       | <i>LL_TIM_OC_SetMode</i>          |
| OC1PE    |       | <i>LL_TIM_OC_DisablePreload</i>   |
|          |       | <i>LL_TIM_OC_EnablePreload</i>    |
|          |       | <i>LL_TIM_OC_IsEnabledPreload</i> |
| OC2CE    |       | <i>LL_TIM_OC_DisableClear</i>     |
|          |       | <i>LL_TIM_OC_EnableClear</i>      |
|          |       | <i>LL_TIM_OC_IsEnabledClear</i>   |
| OC2FE    |       | <i>LL_TIM_OC_DisableFast</i>      |
|          |       | <i>LL_TIM_OC_EnableFast</i>       |
|          |       | <i>LL_TIM_OC_IsEnabledFast</i>    |
| OC2M     |       | <i>LL_TIM_OC_GetMode</i>          |
|          |       | <i>LL_TIM_OC_SetMode</i>          |
| OC2PE    |       | <i>LL_TIM_OC_DisablePreload</i>   |
|          |       | <i>LL_TIM_OC_EnablePreload</i>    |
|          |       | <i>LL_TIM_OC_IsEnabledPreload</i> |
| CCMR2    | CC3S  | <i>LL_TIM_IC_Config</i>           |

| Register | Field  | Function                          |
|----------|--------|-----------------------------------|
| TIMx     | ICx    | <i>LL_TIM_IC_GetActiveInput</i>   |
|          |        | <i>LL_TIM_IC_SetActiveInput</i>   |
|          |        | <i>LL_TIM_OC_ConfigOutput</i>     |
|          | CC4S   | <i>LL_TIM_IC_Config</i>           |
|          |        | <i>LL_TIM_IC_GetActiveInput</i>   |
|          |        | <i>LL_TIM_IC_SetActiveInput</i>   |
|          |        | <i>LL_TIM_OC_ConfigOutput</i>     |
|          | IC3F   | <i>LL_TIM_IC_Config</i>           |
|          |        | <i>LL_TIM_IC_GetFilter</i>        |
|          |        | <i>LL_TIM_IC_SetFilter</i>        |
|          | IC3PSC | <i>LL_TIM_IC_Config</i>           |
|          |        | <i>LL_TIM_IC_GetPrescaler</i>     |
|          |        | <i>LL_TIM_IC_SetPrescaler</i>     |
|          | IC4F   | <i>LL_TIM_IC_Config</i>           |
|          |        | <i>LL_TIM_IC_GetFilter</i>        |
|          |        | <i>LL_TIM_IC_SetFilter</i>        |
|          | IC4PSC | <i>LL_TIM_IC_Config</i>           |
|          |        | <i>LL_TIM_IC_GetPrescaler</i>     |
|          |        | <i>LL_TIM_IC_SetPrescaler</i>     |
|          | OC3CE  | <i>LL_TIM_OC_DisableClear</i>     |
|          |        | <i>LL_TIM_OC_EnableClear</i>      |
|          |        | <i>LL_TIM_OC_IsEnabledClear</i>   |
|          | OC3FE  | <i>LL_TIM_OC_DisableFast</i>      |
|          |        | <i>LL_TIM_OC_EnableFast</i>       |
|          |        | <i>LL_TIM_OC_IsEnabledFast</i>    |
|          | OC3M   | <i>LL_TIM_OC_GetMode</i>          |
|          |        | <i>LL_TIM_OC_SetMode</i>          |
|          | OC3PE  | <i>LL_TIM_OC_DisablePreload</i>   |
|          |        | <i>LL_TIM_OC_EnablePreload</i>    |
|          |        | <i>LL_TIM_OC_IsEnabledPreload</i> |
|          | OC4CE  | <i>LL_TIM_OC_DisableClear</i>     |
|          |        | <i>LL_TIM_OC_EnableClear</i>      |
|          |        | <i>LL_TIM_OC_IsEnabledClear</i>   |
|          | OC4FE  | <i>LL_TIM_OC_DisableFast</i>      |
|          |        | <i>LL_TIM_OC_EnableFast</i>       |
|          |        | <i>LL_TIM_OC_IsEnabledFast</i>    |

| Register | Field | Function                          |
|----------|-------|-----------------------------------|
| CCMR3    | OC4M  | <i>LL_TIM_OC_GetMode</i>          |
|          |       | <i>LL_TIM_OC_SetMode</i>          |
|          | OC4PE | <i>LL_TIM_OC_DisablePreload</i>   |
|          |       | <i>LL_TIM_OC_EnablePreload</i>    |
|          |       | <i>LL_TIM_OC_IsEnabledPreload</i> |
|          | CC5S  | <i>LL_TIM_OC_ConfigOutput</i>     |
|          | CC6S  | <i>LL_TIM_OC_ConfigOutput</i>     |
|          | OC5CE | <i>LL_TIM_OC_DisableClear</i>     |
|          |       | <i>LL_TIM_OC_EnableClear</i>      |
|          |       | <i>LL_TIM_OC_IsEnabledClear</i>   |
|          | OC5FE | <i>LL_TIM_OC_DisableFast</i>      |
|          |       | <i>LL_TIM_OC_EnableFast</i>       |
|          |       | <i>LL_TIM_OC_IsEnabledFast</i>    |
|          | OC5M  | <i>LL_TIM_OC_GetMode</i>          |
|          | OC5M  | <i>LL_TIM_OC_SetMode</i>          |
|          | OC5PE | <i>LL_TIM_OC_DisablePreload</i>   |
|          |       | <i>LL_TIM_OC_EnablePreload</i>    |
|          |       | <i>LL_TIM_OC_IsEnabledPreload</i> |
|          | OC6CE | <i>LL_TIM_OC_DisableClear</i>     |
|          |       | <i>LL_TIM_OC_EnableClear</i>      |
|          |       | <i>LL_TIM_OC_IsEnabledClear</i>   |
|          | OC6FE | <i>LL_TIM_OC_DisableFast</i>      |
|          |       | <i>LL_TIM_OC_EnableFast</i>       |
|          |       | <i>LL_TIM_OC_IsEnabledFast</i>    |
|          | OC6M  | <i>LL_TIM_OC_GetMode</i>          |
|          |       | <i>LL_TIM_OC_SetMode</i>          |
|          | OC6PE | <i>LL_TIM_OC_DisablePreload</i>   |
|          |       | <i>LL_TIM_OC_EnablePreload</i>    |
|          |       | <i>LL_TIM_OC_IsEnabledPreload</i> |
| CCR1     | CCR1  | <i>LL_TIM_IC_GetCaptureCH1</i>    |
|          |       | <i>LL_TIM_OC_GetCompareCH1</i>    |
|          |       | <i>LL_TIM_OC_SetCompareCH1</i>    |
| CCR2     | CCR2  | <i>LL_TIM_IC_GetCaptureCH2</i>    |
|          |       | <i>LL_TIM_OC_GetCompareCH2</i>    |
|          |       | <i>LL_TIM_OC_SetCompareCH2</i>    |
| CCR3     | CCR3  | <i>LL_TIM_IC_GetCaptureCH3</i>    |

| Register | Field    | Function                           |
|----------|----------|------------------------------------|
|          |          | <i>LL_TIM_OC_GetCompareCH3</i>     |
|          |          | <i>LL_TIM_OC_SetCompareCH3</i>     |
| CCR4     | CCR4     | <i>LL_TIM_IC_GetCaptureCH4</i>     |
|          |          | <i>LL_TIM_OC_GetCompareCH4</i>     |
|          |          | <i>LL_TIM_OC_SetCompareCH4</i>     |
| CCR5     | CCR5     | <i>LL_TIM_OC_SetCompareCH5</i>     |
| CNT      | CNT      | <i>LL_TIM_GetCounter</i>           |
|          |          | <i>LL_TIM_SetCounter</i>           |
| CR1      | ARPE     | <i>LL_TIM_DisableARRPreload</i>    |
|          |          | <i>LL_TIM_EnableARRPreload</i>     |
|          |          | <i>LL_TIM_IsEnabledARRPreload</i>  |
|          | CEN      | <i>LL_TIM_DisableCounter</i>       |
|          |          | <i>LL_TIM_EnableCounter</i>        |
|          |          | <i>LL_TIM_IsEnabledCounter</i>     |
|          | CKD      | <i>LL_TIM_GetClockDivision</i>     |
|          |          | <i>LL_TIM_SetClockDivision</i>     |
|          | CMS      | <i>LL_TIM_GetCounterMode</i>       |
|          |          | <i>LL_TIM_SetCounterMode</i>       |
|          | DIR      | <i>LL_TIM_GetCounterMode</i>       |
|          |          | <i>LL_TIM_GetDirection</i>         |
|          |          | <i>LL_TIM_SetCounterMode</i>       |
|          | OPM      | <i>LL_TIM_GetOnePulseMode</i>      |
|          |          | <i>LL_TIM_SetOnePulseMode</i>      |
|          | UDIS     | <i>LL_TIM_DisableUpdateEvent</i>   |
|          |          | <i>LL_TIM_EnableUpdateEvent</i>    |
|          |          | <i>LL_TIM_IsEnabledUpdateEvent</i> |
|          | UIFREMAP | <i>LL_TIM_DisableUIFRemap</i>      |
|          |          | <i>LL_TIM_EnableUIFRemap</i>       |
|          | URS      | <i>LL_TIM_GetUpdateSource</i>      |
|          |          | <i>LL_TIM_SetUpdateSource</i>      |
| CR2      | CCDS     | <i>LL_TIM_CC_GetDMAReqTrigger</i>  |
|          |          | <i>LL_TIM_CC_SetDMAReqTrigger</i>  |
|          | CCPC     | <i>LL_TIM_CC_DisablePreload</i>    |
|          |          | <i>LL_TIM_CC_EnablePreload</i>     |
|          | CCUS     | <i>LL_TIM_CC_SetUpdate</i>         |
|          | MMS      | <i>LL_TIM_SetTriggerOutput</i>     |

| Register | Field | Function                                 |
|----------|-------|------------------------------------------|
| OIS1     | MMS2  | <i>LL_TIM_SetTriggerOutput2</i>          |
|          | OIS1  | <i>LL_TIM_OC_ConfigOutput</i>            |
|          |       | <i>LL_TIM_OC_GetIdleState</i>            |
|          |       | <i>LL_TIM_OC_SetIdleState</i>            |
|          | OIS2  | <i>LL_TIM_OC_ConfigOutput</i>            |
|          |       | <i>LL_TIM_OC_GetIdleState</i>            |
|          |       | <i>LL_TIM_OC_SetIdleState</i>            |
|          | OIS2N | <i>LL_TIM_OC_GetIdleState</i>            |
|          |       | <i>LL_TIM_OC_SetIdleState</i>            |
|          | OIS3  | <i>LL_TIM_OC_ConfigOutput</i>            |
|          |       | <i>LL_TIM_OC_GetIdleState</i>            |
|          |       | <i>LL_TIM_OC_SetIdleState</i>            |
|          | OIS3N | <i>LL_TIM_OC_GetIdleState</i>            |
|          |       | <i>LL_TIM_OC_SetIdleState</i>            |
|          | OIS4  | <i>LL_TIM_OC_ConfigOutput</i>            |
|          |       | <i>LL_TIM_OC_GetIdleState</i>            |
|          |       | <i>LL_TIM_OC_SetIdleState</i>            |
|          | OIS5  | <i>LL_TIM_OC_ConfigOutput</i>            |
|          |       | <i>LL_TIM_OC_GetIdleState</i>            |
|          |       | <i>LL_TIM_OC_SetIdleState</i>            |
|          | OIS6  | <i>LL_TIM_OC_ConfigOutput</i>            |
|          |       | <i>LL_TIM_OC_GetIdleState</i>            |
|          |       | <i>LL_TIM_OC_SetIdleState</i>            |
|          | TI1S  | <i>LL_TIM_IC_DisableXORCombination</i>   |
|          |       | <i>LL_TIM_IC_EnableXORCombination</i>    |
|          |       | <i>LL_TIM_IC_IsEnabledXORCombination</i> |
| DCR      | DBA   | <i>LL_TIM_ConfigDMABurst</i>             |
|          | DBL   | <i>LL_TIM_ConfigDMABurst</i>             |
| DIER     | BIE   | <i>LL_TIM_DisableIT_BRK</i>              |
|          |       | <i>LL_TIM_EnableIT_BRK</i>               |
|          |       | <i>LL_TIM_IsEnabledIT_BRK</i>            |
|          | CC1DE | <i>LL_TIM_DisableDMAReq_CC1</i>          |
|          |       | <i>LL_TIM_EnableDMAReq_CC1</i>           |
|          |       | <i>LL_TIM_IsEnabledDMAReq_CC1</i>        |
|          | CC1IE | <i>LL_TIM_DisableIT_CC1</i>              |
|          |       | <i>LL_TIM_EnableIT_CC1</i>               |

| Register | Field | Function                             |
|----------|-------|--------------------------------------|
| CC2DE    |       | <i>LL_TIM_IsEnabledIT_CC1</i>        |
|          |       | <i>LL_TIM_DisableDMAReq_CC2</i>      |
|          |       | <i>LL_TIM_EnableDMAReq_CC2</i>       |
|          |       | <i>LL_TIM_IsEnabledDMAReq_CC2</i>    |
| CC2IE    |       | <i>LL_TIM_DisableIT_CC2</i>          |
|          |       | <i>LL_TIM_EnableIT_CC2</i>           |
|          |       | <i>LL_TIM_IsEnabledIT_CC2</i>        |
| CC3DE    |       | <i>LL_TIM_DisableDMAReq_CC3</i>      |
|          |       | <i>LL_TIM_EnableDMAReq_CC3</i>       |
|          |       | <i>LL_TIM_IsEnabledDMAReq_CC3</i>    |
| CC3IE    |       | <i>LL_TIM_DisableIT_CC3</i>          |
|          |       | <i>LL_TIM_EnableIT_CC3</i>           |
|          |       | <i>LL_TIM_IsEnabledIT_CC3</i>        |
| CC4DE    |       | <i>LL_TIM_DisableDMAReq_CC4</i>      |
|          |       | <i>LL_TIM_EnableDMAReq_CC4</i>       |
|          |       | <i>LL_TIM_IsEnabledDMAReq_CC4</i>    |
| CC4IE    |       | <i>LL_TIM_DisableIT_CC4</i>          |
|          |       | <i>LL_TIM_EnableIT_CC4</i>           |
|          |       | <i>LL_TIM_IsEnabledIT_CC4</i>        |
| COMDE    |       | <i>LL_TIM_DisableDMAReq_COM</i>      |
|          |       | <i>LL_TIM_EnableDMAReq_COM</i>       |
|          |       | <i>LL_TIM_IsEnabledDMAReq_COM</i>    |
| COMIE    |       | <i>LL_TIM_DisableIT_COM</i>          |
|          |       | <i>LL_TIM_EnableIT_COM</i>           |
|          |       | <i>LL_TIM_IsEnabledIT_COM</i>        |
| TDE      |       | <i>LL_TIM_DisableDMAReq_TRIG</i>     |
|          |       | <i>LL_TIM_EnableDMAReq_TRIG</i>      |
|          |       | <i>LL_TIM_IsEnabledDMAReq_TRIG</i>   |
| TIE      |       | <i>LL_TIM_DisableIT_TRIG</i>         |
|          |       | <i>LL_TIM_EnableIT_TRIG</i>          |
|          |       | <i>LL_TIM_IsEnabledIT_TRIG</i>       |
| UDE      |       | <i>LL_TIM_DisableDMAReq_UPDATE</i>   |
|          |       | <i>LL_TIM_EnableDMAReq_UPDATE</i>    |
|          |       | <i>LL_TIM_IsEnabledDMAReq_UPDATE</i> |
| UIE      |       | <i>LL_TIM_DisableIT_UPDATE</i>       |
|          |       | <i>LL_TIM_EnableIT_UPDATE</i>        |

| Register | Field | Function                               |
|----------|-------|----------------------------------------|
| EGR      | B2G   | <i>LL_TIM_IsEnabledIT_UPDATE</i>       |
|          | BG    | <i>LL_TIM_GenerateEvent_BRK2</i>       |
|          | CC1G  | <i>LL_TIM_GenerateEvent_BRK</i>        |
|          | CC2G  | <i>LL_TIM_GenerateEvent_CC1</i>        |
|          | CC3G  | <i>LL_TIM_GenerateEvent_CC2</i>        |
|          | CC4G  | <i>LL_TIM_GenerateEvent_CC3</i>        |
|          | COMG  | <i>LL_TIM_GenerateEvent_CC4</i>        |
|          | TG    | <i>LL_TIM_GenerateEvent_COM</i>        |
|          | UG    | <i>LL_TIM_GenerateEvent_TRIG</i>       |
| PSC      | PSC   | <i>LL_TIM_SetPrescaler</i>             |
|          |       | <i>LL_TIM_GetPrescaler</i>             |
| RCR      | REP   | <i>LL_TIM_SetRepetitionCounter</i>     |
|          |       | <i>LL_TIM_GetRepetitionCounter</i>     |
| SMCR     | ECE   | <i>LL_TIM_DisableExternalClock</i>     |
|          |       | <i>LL_TIM_EnableExternalClock</i>      |
|          |       | <i>LL_TIM_IsEnabledExternalClock</i>   |
|          |       | <i>LL_TIM_SetClockSource</i>           |
|          | ETF   | <i>LL_TIM_ConfigETR</i>                |
|          | ETP   | <i>LL_TIM_ConfigETR</i>                |
|          | ETPS  | <i>LL_TIM_ConfigETR</i>                |
|          | MSM   | <i>LL_TIM_DisableMasterSlaveMode</i>   |
|          |       | <i>LL_TIM_EnableMasterSlaveMode</i>    |
|          |       | <i>LL_TIM_IsEnabledMasterSlaveMode</i> |
|          | OCCS  | <i>LL_TIM_SetOCRefClearInputSource</i> |
|          | SMS   | <i>LL_TIM_SetClockSource</i>           |
|          |       | <i>LL_TIM_SetEncoderMode</i>           |
|          |       | <i>LL_TIM_SetSlaveMode</i>             |
|          | TS    | <i>LL_TIM_SetTriggerInput</i>          |
| SR       | B2IF  | <i>LL_TIM_ClearFlag_BRK2</i>           |
|          |       | <i>LL_TIM_IsActiveFlag_BRK2</i>        |
|          | BIF   | <i>LL_TIM_ClearFlag_BRK</i>            |
|          |       | <i>LL_TIM_IsActiveFlag_BRK</i>         |
|          | CC1IF | <i>LL_TIM_ClearFlag_CC1</i>            |
|          |       | <i>LL_TIM_IsActiveFlag_CC1</i>         |
|          | CC1OF | <i>LL_TIM_ClearFlag_CC1OVR</i>         |

| Register  | Field    | Function                          |
|-----------|----------|-----------------------------------|
| TIMx_SR   | CC2IF    | <i>LL_TIM_IsActiveFlag_CC1OVR</i> |
|           |          | <i>LL_TIM_ClearFlag_CC2</i>       |
|           |          | <i>LL_TIM_IsActiveFlag_CC2</i>    |
|           | CC2OF    | <i>LL_TIM_ClearFlag_CC2OVR</i>    |
|           |          | <i>LL_TIM_IsActiveFlag_CC2OVR</i> |
|           | CC3IF    | <i>LL_TIM_ClearFlag_CC3</i>       |
|           |          | <i>LL_TIM_IsActiveFlag_CC3</i>    |
|           | CC3OF    | <i>LL_TIM_ClearFlag_CC3OVR</i>    |
|           |          | <i>LL_TIM_IsActiveFlag_CC3OVR</i> |
|           | CC4IF    | <i>LL_TIM_ClearFlag_CC4</i>       |
|           |          | <i>LL_TIM_IsActiveFlag_CC4</i>    |
| TIMx_EGR  | CC4OF    | <i>LL_TIM_ClearFlag_CC4OVR</i>    |
|           |          | <i>LL_TIM_IsActiveFlag_CC4OVR</i> |
|           | CC5IF    | <i>LL_TIM_ClearFlag_CC5</i>       |
|           |          | <i>LL_TIM_IsActiveFlag_CC5</i>    |
|           | CC6IF    | <i>LL_TIM_ClearFlag_CC6</i>       |
|           |          | <i>LL_TIM_IsActiveFlag_CC6</i>    |
|           | COMIF    | <i>LL_TIM_ClearFlag_COM</i>       |
|           |          | <i>LL_TIM_IsActiveFlag_COM</i>    |
|           | TIF      | <i>LL_TIM_ClearFlag_TRIG</i>      |
|           |          | <i>LL_TIM_IsActiveFlag_TRIG</i>   |
| TIMx_DIER | UIF      | <i>LL_TIM_ClearFlag_UPDATE</i>    |
|           |          | <i>LL_TIM_IsActiveFlag_UPDATE</i> |
|           | TIM1_OR  | <i>LL_TIM_SetRemap</i>            |
|           | TIM20_OR | <i>LL_TIM_SetRemap</i>            |
|           | TIM8_OR  | <i>LL_TIM_SetRemap</i>            |

## 82.20 USART

Table 44: Correspondence between USART registers and USART low-layer driver functions

| Register | Field | Function                           |
|----------|-------|------------------------------------|
| BRR      | BRR   | <i>LL_USART_GetBaudRate</i>        |
|          |       | <i>LL_USART_SetBaudRate</i>        |
| CR1      | CMIE  | <i>LL_USART_DisableIT_CM</i>       |
|          |       | <i>LL_USART_EnableIT_CM</i>        |
|          |       | <i>LL_USART_IsEnabledIT_CM</i>     |
|          | DEAT  | <i>LL_USART_GetDEAssertionTime</i> |

| Register | Field | Function                             |
|----------|-------|--------------------------------------|
| DEDT     |       | <i>LL_USART_SetDEAssertionTime</i>   |
|          |       | <i>LL_USART_GetDEDeassertionTime</i> |
|          |       | <i>LL_USART_SetDEDeassertionTime</i> |
| EOBIE    |       | <i>LL_USART_DisableIT_EOB</i>        |
|          |       | <i>LL_USART_EnableIT_EOB</i>         |
|          |       | <i>LL_USART_IsEnabledIT_EOB</i>      |
| IDLEIE   |       | <i>LL_USART_DisableIT_IDLE</i>       |
|          |       | <i>LL_USART_EnableIT_IDLE</i>        |
|          |       | <i>LL_USART_IsEnabledIT_IDLE</i>     |
| M0       |       | <i>LL_USART_ConfigCharacter</i>      |
|          |       | <i>LL_USART_GetDataWidth</i>         |
|          |       | <i>LL_USART_SetDataWidth</i>         |
| M1       |       | <i>LL_USART_ConfigCharacter</i>      |
|          |       | <i>LL_USART_GetDataWidth</i>         |
|          |       | <i>LL_USART_SetDataWidth</i>         |
| MME      |       | <i>LL_USART_DisableMuteMode</i>      |
|          |       | <i>LL_USART_EnableMuteMode</i>       |
|          |       | <i>LL_USART_IsEnabledMuteMode</i>    |
| OVER8    |       | <i>LL_USART_GetOverSampling</i>      |
|          |       | <i>LL_USART_SetOverSampling</i>      |
| PCE      |       | <i>LL_USART_ConfigCharacter</i>      |
|          |       | <i>LL_USART_GetParity</i>            |
|          |       | <i>LL_USART_SetParity</i>            |
| PEIE     |       | <i>LL_USART_DisableIT_PE</i>         |
|          |       | <i>LL_USART_EnableIT_PE</i>          |
|          |       | <i>LL_USART_IsEnabledIT_PE</i>       |
| PS       |       | <i>LL_USART_ConfigCharacter</i>      |
|          |       | <i>LL_USART_GetParity</i>            |
|          |       | <i>LL_USART_SetParity</i>            |
| RE       |       | <i>LL_USART_DisableDirectionRx</i>   |
|          |       | <i>LL_USART_EnableDirectionRx</i>    |
|          |       | <i>LL_USART_GetTransferDirection</i> |
|          |       | <i>LL_USART_SetTransferDirection</i> |
| RTOIE    |       | <i>LL_USART_DisableIT_RTO</i>        |
|          |       | <i>LL_USART_EnableIT_RTO</i>         |
|          |       | <i>LL_USART_IsEnabledIT_RTO</i>      |

| Register | Field   | Function                                                     |
|----------|---------|--------------------------------------------------------------|
|          | RXNEIE  | <a href="#"><code>LL_USART_DisableIT_RXNE</code></a>         |
|          |         | <a href="#"><code>LL_USART_EnableIT_RXNE</code></a>          |
|          |         | <a href="#"><code>LL_USART_IsEnabledIT_RXNE</code></a>       |
|          | TCIE    | <a href="#"><code>LL_USART_DisableIT_TC</code></a>           |
|          |         | <a href="#"><code>LL_USART_EnableIT_TC</code></a>            |
|          |         | <a href="#"><code>LL_USART_IsEnabledIT_TC</code></a>         |
|          | TE      | <a href="#"><code>LL_USART_DisableDirectionTx</code></a>     |
|          |         | <a href="#"><code>LL_USART_EnableDirectionTx</code></a>      |
|          |         | <a href="#"><code>LL_USART_GetTransferDirection</code></a>   |
|          |         | <a href="#"><code>LL_USART_SetTransferDirection</code></a>   |
|          | TXEIE   | <a href="#"><code>LL_USART_DisableIT_TXE</code></a>          |
|          |         | <a href="#"><code>LL_USART_EnableIT_TXE</code></a>           |
|          |         | <a href="#"><code>LL_USART_IsEnabledIT_TXE</code></a>        |
|          | UE      | <a href="#"><code>LL_USART_Disable</code></a>                |
|          |         | <a href="#"><code>LL_USART_Enable</code></a>                 |
|          |         | <a href="#"><code>LL_USART_IsEnabled</code></a>              |
|          | UESM    | <a href="#"><code>LL_USART_DisableInStopMode</code></a>      |
|          |         | <a href="#"><code>LL_USART_EnableInStopMode</code></a>       |
|          |         | <a href="#"><code>LL_USART_IsEnabledInStopMode</code></a>    |
|          | WAKE    | <a href="#"><code>LL_USART_GetWakeUpMethod</code></a>        |
|          |         | <a href="#"><code>LL_USART_SetWakeUpMethod</code></a>        |
| CR2      | ABREN   | <a href="#"><code>LL_USART_DisableAutoBaudRate</code></a>    |
|          |         | <a href="#"><code>LL_USART_EnableAutoBaudRate</code></a>     |
|          |         | <a href="#"><code>LL_USART_IsEnabledAutoBaud</code></a>      |
|          | ABRMODE | <a href="#"><code>LL_USART_GetAutoBaudRateMode</code></a>    |
|          |         | <a href="#"><code>LL_USART_SetAutoBaudRateMode</code></a>    |
|          | ADD     | <a href="#"><code>LL_USART_ConfigNodeAddress</code></a>      |
|          |         | <a href="#"><code>LL_USART_GetNodeAddress</code></a>         |
|          | ADDM7   | <a href="#"><code>LL_USART_ConfigNodeAddress</code></a>      |
|          |         | <a href="#"><code>LL_USART_GetNodeAddressLen</code></a>      |
|          | CLKEN   | <a href="#"><code>LL_USART_ConfigAsyncMode</code></a>        |
|          |         | <a href="#"><code>LL_USART_ConfigHalfDuplexMode</code></a>   |
|          |         | <a href="#"><code>LL_USART_ConfigIrdaMode</code></a>         |
|          |         | <a href="#"><code>LL_USART_ConfigLINMode</code></a>          |
|          |         | <a href="#"><code>LL_USART_ConfigMultiProcessMode</code></a> |
|          |         | <a href="#"><code>LL_USART_ConfigSmartcardMode</code></a>    |

| Register | Field    | Function                                               |
|----------|----------|--------------------------------------------------------|
|          |          | <a href="#"><i>LL_USART_ConfigSyncMode</i></a>         |
|          |          | <a href="#"><i>LL_USART_DisableSCLKOutput</i></a>      |
|          |          | <a href="#"><i>LL_USART_EnableSCLKOutput</i></a>       |
|          |          | <a href="#"><i>LL_USART_IsEnabledSCLKOutput</i></a>    |
|          | CPHA     | <a href="#"><i>LL_USART_ConfigClock</i></a>            |
|          |          | <a href="#"><i>LL_USART_GetClockPhase</i></a>          |
|          |          | <a href="#"><i>LL_USART_SetClockPhase</i></a>          |
|          | CPOL     | <a href="#"><i>LL_USART_ConfigClock</i></a>            |
|          |          | <a href="#"><i>LL_USART_GetClockPolarity</i></a>       |
|          |          | <a href="#"><i>LL_USART_SetClockPolarity</i></a>       |
|          | DATAINV  | <a href="#"><i>LL_USART_GetBinaryDataLogic</i></a>     |
|          |          | <a href="#"><i>LL_USART_SetBinaryDataLogic</i></a>     |
|          | LBCL     | <a href="#"><i>LL_USART_ConfigClock</i></a>            |
|          |          | <a href="#"><i>LL_USART_GetLastClkPulseOutput</i></a>  |
|          |          | <a href="#"><i>LL_USART_SetLastClkPulseOutput</i></a>  |
|          | LBDIE    | <a href="#"><i>LL_USART_DisableIT_LBD</i></a>          |
|          |          | <a href="#"><i>LL_USART_EnableIT_LBD</i></a>           |
|          |          | <a href="#"><i>LL_USART_IsEnabledIT_LBD</i></a>        |
|          | LBDL     | <a href="#"><i>LL_USART_GetLINBrkDetectionLen</i></a>  |
|          |          | <a href="#"><i>LL_USART_SetLINBrkDetectionLen</i></a>  |
|          | LINEN    | <a href="#"><i>LL_USART_ConfigAsyncMode</i></a>        |
|          |          | <a href="#"><i>LL_USART_ConfigHalfDuplexMode</i></a>   |
|          |          | <a href="#"><i>LL_USART_ConfigIrdaMode</i></a>         |
|          |          | <a href="#"><i>LL_USART_ConfigLINMode</i></a>          |
|          |          | <a href="#"><i>LL_USART_ConfigMultiProcessMode</i></a> |
|          |          | <a href="#"><i>LL_USART_ConfigSmartcardMode</i></a>    |
|          |          | <a href="#"><i>LL_USART_ConfigSyncMode</i></a>         |
|          |          | <a href="#"><i>LL_USART_DisableLIN</i></a>             |
|          |          | <a href="#"><i>LL_USART_EnableLIN</i></a>              |
|          |          | <a href="#"><i>LL_USART_IsEnabledLIN</i></a>           |
|          | MSBFIRST | <a href="#"><i>LL_USART_GetTransferBitOrder</i></a>    |
|          |          | <a href="#"><i>LL_USART_SetTransferBitOrder</i></a>    |
|          | RTOEN    | <a href="#"><i>LL_USART_DisableRxTimeout</i></a>       |
|          |          | <a href="#"><i>LL_USART_EnableRxTimeout</i></a>        |
|          |          | <a href="#"><i>LL_USART_IsEnabledRxTimeout</i></a>     |
|          | RXINV    | <a href="#"><i>LL_USART_GetRXPinLevel</i></a>          |

| Register | Field | Function                                                 |
|----------|-------|----------------------------------------------------------|
| CR3      | STOP  | <a href="#"><i>LL_USART_SetRXPinLevel</i></a>            |
|          |       | <a href="#"><i>LL_USART_ConfigCharacter</i></a>          |
|          |       | <a href="#"><i>LL_USART_ConfigIrdaMode</i></a>           |
|          |       | <a href="#"><i>LL_USART_ConfigLINMode</i></a>            |
|          |       | <a href="#"><i>LL_USART_ConfigSmartcardMode</i></a>      |
|          |       | <a href="#"><i>LL_USART_GetStopBitsLength</i></a>        |
|          |       | <a href="#"><i>LL_USART_SetStopBitsLength</i></a>        |
|          | SWAP  | <a href="#"><i>LL_USART_GetTXRXSwap</i></a>              |
|          |       | <a href="#"><i>LL_USART_SetTXRXSwap</i></a>              |
|          | TXINV | <a href="#"><i>LL_USART_GetTXPinLevel</i></a>            |
|          |       | <a href="#"><i>LL_USART_SetTXPinLevel</i></a>            |
|          | CTSE  | <a href="#"><i>LL_USART_DisableCTSHWFlowCtrl</i></a>     |
|          |       | <a href="#"><i>LL_USART_EnableCTSHWFlowCtrl</i></a>      |
|          |       | <a href="#"><i>LL_USART_GetHWFlowCtrl</i></a>            |
|          |       | <a href="#"><i>LL_USART_SetHWFlowCtrl</i></a>            |
|          | CTSIE | <a href="#"><i>LL_USART_DisableIT_CTS</i></a>            |
|          |       | <a href="#"><i>LL_USART_EnableIT_CTS</i></a>             |
|          |       | <a href="#"><i>LL_USART_IsEnabledIT_CTS</i></a>          |
|          | DDRE  | <a href="#"><i>LL_USART_DisableDMADeactOnRxErr</i></a>   |
|          |       | <a href="#"><i>LL_USART_EnableDMADeactOnRxErr</i></a>    |
|          |       | <a href="#"><i>LL_USART_IsEnabledDMADeactOnRxErr</i></a> |
|          | DEM   | <a href="#"><i>LL_USART_DisableDEMode</i></a>            |
|          |       | <a href="#"><i>LL_USART_EnableDEMode</i></a>             |
|          |       | <a href="#"><i>LL_USART_IsEnabledDEMode</i></a>          |
|          | DEP   | <a href="#"><i>LL_USART_GetDESignalPolarity</i></a>      |
|          |       | <a href="#"><i>LL_USART_SetDESignalPolarity</i></a>      |
|          | DMAR  | <a href="#"><i>LL_USART_DisableDMAReq_RX</i></a>         |
|          |       | <a href="#"><i>LL_USART_EnableDMAReq_RX</i></a>          |
|          |       | <a href="#"><i>LL_USART_IsEnabledDMAReq_RX</i></a>       |
|          | DMAT  | <a href="#"><i>LL_USART_DisableDMAReq_TX</i></a>         |
|          |       | <a href="#"><i>LL_USART_EnableDMAReq_TX</i></a>          |
|          |       | <a href="#"><i>LL_USART_IsEnabledDMAReq_TX</i></a>       |
|          | EIE   | <a href="#"><i>LL_USART_DisableIT_ERROR</i></a>          |
|          |       | <a href="#"><i>LL_USART_EnableIT_ERROR</i></a>           |
|          |       | <a href="#"><i>LL_USART_IsEnabledIT_ERROR</i></a>        |
|          | HDSEL | <a href="#"><i>LL_USART_ConfigAsyncMode</i></a>          |

| Register | Field | Function                                            |
|----------|-------|-----------------------------------------------------|
|          |       | <a href="#">LL_USART_ConfigHalfDuplexMode</a>       |
|          |       | <a href="#">LL_USART_ConfigIrdaMode</a>             |
|          |       | <a href="#">LL_USART_ConfigLINMode</a>              |
|          |       | <a href="#">LL_USART_ConfigMultiProcessMode</a>     |
|          |       | <a href="#">LL_USART_ConfigSmartcardMode</a>        |
|          |       | <a href="#">LL_USART_ConfigSyncMode</a>             |
|          |       | <a href="#">LL_USART_DisableHalfDuplex</a>          |
|          |       | <a href="#">LL_USART_EnableHalfDuplex</a>           |
|          |       | <a href="#">LL_USART_IsEnabledHalfDuplex</a>        |
|          |       |                                                     |
| IREN     |       | <a href="#">LL_USART_ConfigAsyncMode</a>            |
|          |       | <a href="#">LL_USART_ConfigHalfDuplexMode</a>       |
|          |       | <a href="#">LL_USART_ConfigIrdaMode</a>             |
|          |       | <a href="#">LL_USART_ConfigLINMode</a>              |
|          |       | <a href="#">LL_USART_ConfigMultiProcessMode</a>     |
|          |       | <a href="#">LL_USART_ConfigSyncMode</a>             |
|          |       | <a href="#">LL_USART_DisableIrda</a>                |
|          |       | <a href="#">LL_USART_EnableIrda</a>                 |
|          |       | <a href="#">LL_USART_IsEnabledIrda</a>              |
|          |       |                                                     |
| IRLP     |       | <a href="#">LL_USART_GetIrdaPowerMode</a>           |
|          |       | <a href="#">LL_USART_SetIrdaPowerMode</a>           |
| NACK     |       | <a href="#">LL_USART_DisableSmartcardNACK</a>       |
|          |       | <a href="#">LL_USART_EnableSmartcardNACK</a>        |
|          |       | <a href="#">LL_USART_IsEnabledSmartcardNACK</a>     |
| ONEBIT   |       | <a href="#">LL_USART_DisableOneBitSamp</a>          |
|          |       | <a href="#">LL_USART_EnableOneBitSamp</a>           |
|          |       | <a href="#">LL_USART_IsEnabledOneBitSamp</a>        |
| OVRDIS   |       | <a href="#">LL_USART_DisableOverrunDetect</a>       |
|          |       | <a href="#">LL_USART_EnableOverrunDetect</a>        |
|          |       | <a href="#">LL_USART_IsEnabledOverrunDetect</a>     |
| RTSE     |       | <a href="#">LL_USART_DisableRTSHWFlowCtrl</a>       |
|          |       | <a href="#">LL_USART_EnableRTSHWFlowCtrl</a>        |
|          |       | <a href="#">LL_USART_GetHWFlowCtrl</a>              |
|          |       | <a href="#">LL_USART_SetHWFlowCtrl</a>              |
| SCARCNT  |       | <a href="#">LL_USART_GetSmartcardAutoRetryCount</a> |
|          |       | <a href="#">LL_USART_SetSmartcardAutoRetryCount</a> |
| SCEN     |       | <a href="#">LL_USART_ConfigAsyncMode</a>            |

| Register | Field  | Function                                               |
|----------|--------|--------------------------------------------------------|
|          |        | <a href="#"><i>LL_USART_ConfigHalfDuplexMode</i></a>   |
|          |        | <a href="#"><i>LL_USART_ConfigIrdaMode</i></a>         |
|          |        | <a href="#"><i>LL_USART_ConfigLINMode</i></a>          |
|          |        | <a href="#"><i>LL_USART_ConfigMultiProcessMode</i></a> |
|          |        | <a href="#"><i>LL_USART_ConfigSmartcardMode</i></a>    |
|          |        | <a href="#"><i>LL_USART_ConfigSyncMode</i></a>         |
|          |        | <a href="#"><i>LL_USART_DisableSmartcard</i></a>       |
|          |        | <a href="#"><i>LL_USART_EnableSmartcard</i></a>        |
|          |        | <a href="#"><i>LL_USART_IsEnabledSmartcard</i></a>     |
|          |        |                                                        |
| WUFIE    |        | <a href="#"><i>LL_USART_DisableIT_WKUP</i></a>         |
|          |        | <a href="#"><i>LL_USART_EnableIT_WKUP</i></a>          |
|          |        | <a href="#"><i>LL_USART_IsEnabledIT_WKUP</i></a>       |
| WUS      |        | <a href="#"><i>LL_USART_GetWKUPType</i></a>            |
|          |        | <a href="#"><i>LL_USART_SetWKUPType</i></a>            |
| GTPR     | GT     | <a href="#"><i>LL_USART_GetSmartcardGuardTime</i></a>  |
|          |        | <a href="#"><i>LL_USART_SetSmartcardGuardTime</i></a>  |
|          | PSC    | <a href="#"><i>LL_USART_GetIrdaPrescaler</i></a>       |
|          |        | <a href="#"><i>LL_USART_GetSmartcardPrescaler</i></a>  |
|          |        | <a href="#"><i>LL_USART_SetIrdaPrescaler</i></a>       |
|          |        | <a href="#"><i>LL_USART_SetSmartcardPrescaler</i></a>  |
| ICR      | CMCF   | <a href="#"><i>LL_USART_ClearFlag_CM</i></a>           |
|          | CTSCF  | <a href="#"><i>LL_USART_ClearFlag_nCTS</i></a>         |
|          | EOBCF  | <a href="#"><i>LL_USART_ClearFlag_EOB</i></a>          |
|          | FECF   | <a href="#"><i>LL_USART_ClearFlag_FE</i></a>           |
|          | IDLECF | <a href="#"><i>LL_USART_ClearFlag_IDLE</i></a>         |
|          | LBDCF  | <a href="#"><i>LL_USART_ClearFlag_LBD</i></a>          |
|          | NCF    | <a href="#"><i>LL_USART_ClearFlag_NE</i></a>           |
|          | ORECF  | <a href="#"><i>LL_USART_ClearFlag_ORE</i></a>          |
|          | PECF   | <a href="#"><i>LL_USART_ClearFlag_PE</i></a>           |
|          | RTOCF  | <a href="#"><i>LL_USART_ClearFlag_RTO</i></a>          |
|          | TCCF   | <a href="#"><i>LL_USART_ClearFlag_TC</i></a>           |
|          | WUCF   | <a href="#"><i>LL_USART_ClearFlag_WKUP</i></a>         |
| ISR      | ABRE   | <a href="#"><i>LL_USART_IsActiveFlag_ABRE</i></a>      |
|          | ABRF   | <a href="#"><i>LL_USART_IsActiveFlag_ABR</i></a>       |
|          | BUSY   | <a href="#"><i>LL_USART_IsActiveFlag_BUSY</i></a>      |
|          | CMF    | <a href="#"><i>LL_USART_IsActiveFlag_CM</i></a>        |

| Register | Field | Function                             |
|----------|-------|--------------------------------------|
|          | CTS   | <i>LL_USART_IsActiveFlag_CTS</i>     |
|          | CTSIF | <i>LL_USART_IsActiveFlag_nCTS</i>    |
|          | EOBF  | <i>LL_USART_IsActiveFlag_EOB</i>     |
|          | FE    | <i>LL_USART_IsActiveFlag_FE</i>      |
|          | IDLE  | <i>LL_USART_IsActiveFlag_IDLE</i>    |
|          | LBDF  | <i>LL_USART_IsActiveFlag_LBD</i>     |
|          | NF    | <i>LL_USART_IsActiveFlag_NE</i>      |
|          | ORE   | <i>LL_USART_IsActiveFlag_ORE</i>     |
|          | PE    | <i>LL_USART_IsActiveFlag_PE</i>      |
|          | REACK | <i>LL_USART_IsActiveFlag_REACK</i>   |
|          | RTOF  | <i>LL_USART_IsActiveFlag_RTO</i>     |
|          | RWU   | <i>LL_USART_IsActiveFlag_RWU</i>     |
|          | RXNE  | <i>LL_USART_IsActiveFlag_RXNE</i>    |
|          | SBKF  | <i>LL_USART_IsActiveFlag_SBK</i>     |
|          | TC    | <i>LL_USART_IsActiveFlag_TC</i>      |
|          | TEACK | <i>LL_USART_IsActiveFlag_TEACK</i>   |
|          | TXE   | <i>LL_USART_IsActiveFlag_TXE</i>     |
|          | WUF   | <i>LL_USART_IsActiveFlag_WKUP</i>    |
| RDR      | RDR   | <i>LL_USART_DMA_GetRegAddr</i>       |
|          |       | <i>LL_USART_ReceiveData8</i>         |
|          |       | <i>LL_USART_ReceiveData9</i>         |
| RQR      | ABRRQ | <i>LL_USART_RequestAutoBaudRate</i>  |
|          | MMRQ  | <i>LL_USART_RequestEnterMuteMode</i> |
|          | RXFRQ | <i>LL_USART_RequestRxDataFlush</i>   |
|          | SBKRQ | <i>LL_USART_RequestBreakSending</i>  |
|          | TXFRQ | <i>LL_USART_RequestTxDataFlush</i>   |
| RTOR     | BLEN  | <i>LL_USART_GetBlockLength</i>       |
|          |       | <i>LL_USART_SetBlockLength</i>       |
|          | RTO   | <i>LL_USART_GetRxTimeout</i>         |
|          |       | <i>LL_USART_SetRxTimeout</i>         |
| TDR      | TDR   | <i>LL_USART_DMA_GetRegAddr</i>       |
|          |       | <i>LL_USART_TransmitData8</i>        |
|          |       | <i>LL_USART_TransmitData9</i>        |

## 82.21 WWDG

Table 45: Correspondence between WWDG registers and WWDG low-layer driver functions

| Register | Field | Function                          |
|----------|-------|-----------------------------------|
| CFR      | EWI   | <i>LL_WWDG_EnableIT_EWKUP</i>     |
|          |       | <i>LL_WWDG_IsEnabledIT_EWKUP</i>  |
|          | W     | <i>LL_WWDG_GetWindow</i>          |
|          |       | <i>LL_WWDG_SetWindow</i>          |
|          | WDGTB | <i>LL_WWDG_GetPrescaler</i>       |
|          |       | <i>LL_WWDG_SetPrescaler</i>       |
| CR       | T     | <i>LL_WWDG_GetCounter</i>         |
|          |       | <i>LL_WWDG_SetCounter</i>         |
|          | WDGA  | <i>LL_WWDG_Enable</i>             |
|          |       | <i>LL_WWDG_IsEnabled</i>          |
| SR       | EWIF  | <i>LL_WWDG_ClearFlag_EWKUP</i>    |
|          |       | <i>LL_WWDG_IsActiveFlag_EWKUP</i> |

**General subjects****Why should I use the HAL drivers?**

There are many advantages in using the HAL drivers:

- Ease of use: you can use the HAL drivers to configure and control any peripheral embedded within your STM32 MCU without prior in-depth knowledge of the product.
- HAL drivers provide intuitive and ready-to-use APIs to configure the peripherals and support polling, interrupt and DMA programming model to accommodate all application requirements, thus allowing the end-user to build a complete application by calling a few APIs.
- Higher level of abstraction than a standard peripheral library allowing to transparently manage:
  - Data transfers and processing using blocking mode (polling) or non-blocking mode (interrupt or DMA)
  - Error management through peripheral error detection and timeout mechanism.
- Generic architecture speeding up initialization and porting, thus allowing customers to focus on innovation.
- Generic set of APIs with full compatibility across the STM32 series/lines, to ease the porting task between STM32 MCUs.
- The APIs provided within the HAL drivers are feature-oriented and do not require in-depth knowledge of peripheral operation.
- The APIs provided are modular. They include initialization, IO operation and control functions. The end-user has to call init function, then start the process by calling one IO operation functions (write, read, transmit, receive, ...). Most of the peripherals have the same architecture.
- The number of functions required to build a complete and useful application is very reduced. As an example, to build a UART communication process, the user only has to call HAL\_UART\_Init() then HAL\_UART\_Transmit() or HAL\_UART\_Receive().

**Which STM32F3 devices are supported by the HAL drivers?**

The HAL drivers are developed to support all STM32F3 devices. To ensure compatibility between all devices and portability with others series and lines, the API is split into the generic and the extension APIs . For more details, please refer to [Table 5: "List of devices supported by HAL drivers"](#).

**What is the cost of using HAL drivers in term of code size and performance?**

Like generic architecture drivers, the HAL drivers may induce firmware overhead.

This is due to the high abstraction level and ready-to-use APIs which allow data transfers, errors management and offloads the user application from implementation details.

**Architecture****How many files should I modify to configure the HAL drivers?**

Only one file needs to be modified: `stm32f3xx_hal_conf.h`. You can modify this file by disabling unused modules, or adjusting some parameters (i.e. HSE value, System configuration, ...)

A template is provided in the HAL driver folders (stm32f3xx\_hal\_conf\_template.h).

### Which header files should I include in my application to use the HAL drivers?

Only stm32f3xx\_hal.h file has to be included.

### What is the difference between stm32f3xx\_hal\_ppp.c/h and stm32f3xx\_hal\_ppp\_ex.c/h?

The HAL driver architecture supports common features across STM32 series/lines. To support specific features, the drivers are split into two groups.

- The generic APIs (stm32f3xx\_hal\_ppp.c): It includes the common set of APIs across all the STM32 product lines
- The extension APIs (stm32f3xx\_hal\_ppp\_ex.c): It includes the specific APIs for specific device part number or family.

### Initialization and I/O operation functions

#### How do I configure the system clock?

Unlike the standard library, the system clock configuration is not performed in CMSIS drivers file (system\_stm32f3xx.c) but in the main user application by calling the two main functions, HAL\_RCC\_OscConfig() and HAL\_RCC\_ClockConfig(). It can be modified in any user application section.

#### What is the purpose of the *PPP\_HandleTypeDef \*pHandle* structure located in each driver in addition to the Initialization structure

*PPP\_HandleTypeDef \*pHandle* is the main structure implemented in the HAL drivers. It handles the peripheral configuration and registers, and embeds all the structures and variables required to follow the peripheral device flow (pointer to buffer, Error code, State,...)

However, this structure is not required to service peripherals such as GPIO, SYSTICK, PWR, and RCC.

#### What is the purpose of HAL\_PPP\_MspInit() and HAL\_PPP\_MspDeInit() functions?

These function are called within HAL\_PPP\_Init() and HAL\_PPP\_DeInit(), respectively. They are used to perform the low level Initialization/de-initialization related to the additional hardware resources (RCC, GPIO, NVIC and DMA).

These functions are declared in stm32f3xx\_hal\_msp.c. A template is provided in the HAL driver folders (stm32f3xx\_hal\_msp\_template.c).

#### When and how should I use callbacks functions (functions declared with the attribute \_\_weak)?

Use callback functions for the I/O operations used in DMA or interrupt mode. The PPP process complete callbacks are called to inform the user about process completion in real-time event mode (interrupts).

The Errors callbacks are called when a processing error occurs in DMA or interrupt mode. These callbacks are customized by the user to add user proprietary code. They can be declared in the application. Note that the same process completion callbacks are used for DMA and interrupt mode.

### **Is it mandatory to use HAL\_Init() function at the beginning of the user application?**

It is mandatory to use HAL\_Init() function to enable the system configuration (Prefetch, Data instruction cache,...), configure the systTick and the NVIC priority grouping and the hardware low level initialization.

The SysTick configuration shall be adjusted by calling **HAL\_RCC\_ClockConfig()** function, to obtain 1 ms whatever the system clock.

### **Why do I need to configure the SysTick timer to use the HAL drivers?**

The SysTick timer is configured to be used to generate variable increments by calling **HAL\_IncTick()** function in Systick ISR and retrieve the value of this variable by calling **HAL\_GetTick()** function.

The call **HAL\_GetTick()** function is mandatory when using HAL drivers with Polling Process or when using **HAL\_Delay()**.

### **Why is the SysTick timer configured to have 1 ms?**

This is mandatory to ensure correct IO operation in particular for polling mode operation where the 1 ms is required as timebase.

### **Could HAL\_Delay() function block my application under certain conditions?**

Care must be taken when using **HAL\_Delay()** since this function provides accurate delay based on a variable incremented in SysTick ISR. This implies that if **HAL\_Delay()** is called from a peripheral ISR process, then the SysTick interrupt must have higher priority (numerically lower) than the peripheral interrupt, otherwise the caller ISR process will be blocked. Use **HAL\_NVIC\_SetPriority()** function to change the SysTick interrupt priority.

### **What programming model sequence should I follow to use HAL drivers ?**

Follow the sequence below to use the APIs provided in the HAL drivers:

1. Call **HAL\_Init()** function to initialize the system (data cache, NVIC priority,...).
2. Initialize the system clock by calling **HAL\_RCC\_OscConfig()** followed by **HAL\_RCC\_ClockConfig()**.
3. Add **HAL\_IncTick()** function under **SysTick\_Handler()** ISR function to enable polling process when using **HAL\_Delay()** function
4. Start initializing your peripheral by calling **HAL\_PPP\_Init()**.
5. Implement the hardware low level initialization (Peripheral clock, GPIO, DMA,..) by calling **HAL\_PPP\_MspInit()** in **stm32f3xx\_hal\_msp.c**
6. Start your process operation by calling IO operation functions.

### **What is the purpose of HAL\_PPP\_IRQHandler() function and when should I use it?**

**HAL\_PPP\_IRQHandler()** is used to handle interrupt process. It is called under **PPP\_IRQHandler()** function in **stm32f3xx\_it.c**. In this case, the end-user has to implement only the callbacks functions (prefixed by \_\_weak) to perform the appropriate action when an interrupt is detected. Advanced users can implement their own code in **PPP\_IRQHandler()** without calling **HAL\_PPP\_IRQHandler()**.

### **Can I use directly the macros defined in **stm32f3xx\_hal\_ppp.h** ?**

Yes, you can: a set of macros is provided with the APIs. They allow accessing directly some specific features using peripheral flags.

**Where must PPP\_HandleTypeDef structure peripheral handler be declared?**

PPP\_HandleTypeDef structure peripheral handler must be declared as a global variable, so that all the structure fields are set to 0 by default. In this way, the peripheral handler default state are set to HAL\_PPP\_STATE\_RESET, which is the default state for each peripheral after a system reset.

## 84 Revision history

Table 46: Document revision history

| Date        | Revision | Changes                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 05-Dec-2014 | 1        | Initial release.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 04-May-2015 | 2        | <p>Updated Common macros in <a href="#">Section 2.9: "HAL common resources"</a>.</p> <p>Updated <a href="#">Figure 7: "HAL driver model"</a>.</p> <p>Updated <a href="#">Section 27.1.1: "I2S Extended features"</a>.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 19-Jan-2016 | 3        | <p>Added LSE_STARTUP_TIMEOUT in <a href="#">Table 11: "Define statements used for HAL configuration"</a></p> <p>Performed HAL API alignment (macros/functions/constants renaming).</p> <p>Updated <a href="#">Section 7: "HAL ADC Extension Driver"</a>:</p> <ul style="list-style-type: none"> <li>Added HAL_ADCEx-RegularStop(), HAL_ADCEx-RegularStop_IT() and HAL_ADCEx-RegularStop_DMA() functions to perform ADC group regular conversion stop while ADC group injected can remain with conversion on going.</li> <li>Added HAL_ADCEx_LevelOutOfWindow2Callback() and HAL_ADCEx_LevelOutOfWindow3Callback() functions.</li> <li>Updated ADC multimode for devices with several ADC instances: Mixed configuration are now taken into account: ADC group regular in multimode, ADC group injected in independent mode (or the opposite).</li> <li>Updated ADC group injected use-case when the auto-wait low-power feature is used.</li> </ul> <p>Updated <a href="#">Section 8: "HAL CAN Generic Driver"</a>: modified CanTxMsgTypeDef/CanRxMsgTypeDef structures Data field.</p> <p>Updated <a href="#">Section 9: "HAL CEC Generic Driver"</a>:</p> <ul style="list-style-type: none"> <li>Splitted HAL_CEC_ErrorTypeDef structure into separate define statements.</li> <li>Added definitions of CEC flags (CEC_FLAG_TXACKE,...).</li> <li>Add definitions of CEC interrupts (CEC_IT_TXACKE,...).</li> <li>Renamed CEC_ISR_XXX into CEC_FLAG_XXX.</li> <li>Renamed CEC_IER_XXX into CEC_IT_XXX.</li> <li>Added new API HAL_CEC_GetReceivedFrameSize to get the size of the received frame.</li> </ul> <p>Updated <a href="#">Section 12: "HAL CORTEX Generic Driver"</a>:</p> <ul style="list-style-type: none"> <li>Replaced __HAL_CORTEX_SYSTICKCLK_CONFIG macro by HAL_SYSTICK_CLKSourceConfig() function.</li> <li>Added new CORTEX MPU APIs: HAL_MPUCfgRegion(), HAL_MPUDisable() and HAL_MPUEnable().</li> <li>Added APIs to manage set/reset of SLEEPONEXIT and SEVONPEND bits in SCR register.</li> </ul> <p>Updated <a href="#">Section 15: "HAL DAC Generic Driver"</a>: added DAC_OUTPUTSWITCH_ENABLE constant.</p> <p>Updated <a href="#">Section 20: "HAL FLASH Extension Driver"</a>: added FLASH API HAL_FLASHEx_OBGetUserData() to get the value saved in User data option byte.</p> <p>Updated <a href="#">Section 21: "HAL GPIO Generic Driver"</a>: updated GPIO Output Speed naming to ensure HAL full compatibility.</p> <p>Updated <a href="#">Section 28: "HAL IRDA Generic Driver"</a>:</p> <ul style="list-style-type: none"> <li>Implemented the __HAL_UART_FLUSH_DRREGISTER macro, which is required by the In-Application Programming (IAP) using USART.</li> </ul> |

| Date        | Revision         | Changes                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 19-Jan-2016 | 3<br>(continued) | <p>Updated <a href="#">Section 40: "HAL RCC Generic Driver"</a>:</p> <ul style="list-style-type: none"> <li>Renamed __HAL_RCC_MCO_CONFIG into __HAL_RCC_MCO1_CONFIG.</li> </ul> <p>Updated <a href="#">Section 41: "HAL RCC Extension Driver"</a>: updated RCC API functions to add HAL_RCCEx_GetPeriphCLKFreq interface.</p> <p>Updated <a href="#">Section 42: "HAL RTC Generic Driver"</a>:</p> <ul style="list-style-type: none"> <li>Updated definition of __HAL_RTC_TAMPER_TIMESTAMP_EXTI_GET_FLAG.</li> </ul> <p>Aligned different implementations of HAL_RTC_XXIRQHandler(). Updated <a href="#">Section 44: "HAL SDADC Generic Driver"</a>: added new __HAL_SDADC_ENABLE_IT(), __HAL_SDADC_GET_IT_SOURCE() and __HAL_SDADC_GET_FLAG() macros. Updated <a href="#">Section 45: "HAL SMARTCARD Generic Driver"</a>:</p> <ul style="list-style-type: none"> <li>Implemented the __HAL_UART_FLUSH_DRREGISTER macro, which is required by the In-Application Programming (IAP) using USART.</li> <li>Added missing IDLE flag management.</li> </ul> <p>Updated <a href="#">Section 48: "HAL SPI Generic Driver"</a>: added HAL_SPI_Get Error() function.</p> <p>Updated <a href="#">Section 51: "HAL TIM Generic Driver"</a>:</p> <ul style="list-style-type: none"> <li>Removed HAL_TIM_SlaveConfigSynchronization_DMA() from HAL_TIM API functions.</li> <li>Added TIM edge modification macros: TIM_SET_CAPTUREPOLARITY(), TIM_RESET_CAPTUREPOLARITY() and __HAL_TIM_SET_CAPTUREPOLARITY.</li> <li>Added URS_ENABLE, URS_DISABLE macros.</li> <li>Added new HAL_TIM_SlaveConfigSynchronization_IT() function to handle the trigger interrupt activation.</li> </ul> <p>Updated <a href="#">Section 54: "HAL UART Generic Driver"</a> and <a href="#">Section 56: "HAL USART Generic Driver"</a>:</p> <ul style="list-style-type: none"> <li>Implemented the __HAL_UART_FLUSH_DRREGISTER macro, which is required by the In-Application Programming (IAP) using the USART.</li> </ul> <p>Updated IT macro management in <a href="#">Section 58: "HAL WWDG Generic Driver"</a>.</p> |

| Date        | Revision | Changes                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 06-Jun-2016 | 4        | <p>Changed GPIO_SPEED_LOW, GPIO_SPEED_MEDIUM, GPIO_SPEED_HIGH into GPIO_SPEED_FREQ_LOW, GPIO_SPEED_FREQ_MEDIUM, GPIO_SPEED_FREQ_HIGH in section <a href="#">Section 2.11.2: "GPIOs"</a>.</p> <p><b>HAL generic:</b></p> <ul style="list-style-type: none"> <li>• Updated HAL driver compliancy with MISRA C 2004 rules.</li> <li>• Updated HAL weak empty callbacks to prevent unused argument compilation warnings.</li> </ul> <p><b>HAL:</b> changed uwTick to global to allow overwrite of HAL_IncTick().</p> <p><b>HAL COMP:</b> modified COMP_INVERTINGINPUT_SELECTION() macro as COMP inverting inputs selection, depends on COMPx instance.</p> <p><b>HAL DMA:</b></p> <ul style="list-style-type: none"> <li>• Added __HAL_DMA_GET_COUNTER() macro that returns the number of remaining data units in the current DMA Channel transfer.</li> <li>• Provided new function HAL_DMA_Abort_IT() to abort current DMA transfer under interrupt mode without polling for DMA enable bit.</li> </ul> <p><b>HAL GPIO:</b> Added macros to manage Fast Mode Plus on GPIOs.</p> <p><b>HAL I2C:</b></p> <ul style="list-style-type: none"> <li>• Aligned I2C driver with new state machine definition.</li> <li>• Updated __HAL_I2C_DISABLE_IT macro.</li> <li>• Added support for repeated start feature in multimaster mode (this feature allows the master to maintain I2C communications with slave).</li> <li>• Modified HAL_I2C_Master_Transmit to allow sending data of size 0.</li> <li>• Updated DMA Abort management: used new HAL_DMA_Abort() function and called HAL_I2C_ErrorCallback() when errors occur during DMA transfer.</li> </ul> <p><b>HAL I2S:</b> removed support for I2S full-duplex feature on STM32F301x8 device.</p> <p><b>HAL RCC:</b></p> <ul style="list-style-type: none"> <li>• Optimized HAL_RCC_ClockConfig() and HAL_RCCEx_PeriphCLKConfig functions.</li> <li>• Updated HAL_RCC_OscConfig() function (Reset HSEON/LSEON and HSEBYP/LSEBYP bits before configuring the HSE/LSE).</li> <li>• Modified AHBPrescTable and APBPrescTable in HAL.</li> <li>• Removed RCC_CFGR_PLLNODIV bit definition from STM32F358xx, STM32F303xC and STM32F302xC devices.</li> <li>• Removed RCC_CSR_VREGRSTF bit definition in RCC_CSR register for STM32F303xC and STM32F303xE devices.</li> <li>• Removed USART2 and USART3 clock switch in RCC_CFGR3 register not supported by STM32F303x8, STM32F334x8 and STM32F328xx devices and for STM32F301x8, STM32F302x8 and STM32F318xx devices.</li> <li>• Removed RCC_CSR_V18PWRRSTF bit definition in RCC_CSR register not supported by STM32F318xx, STM32F328xx, STM32F358xx, STM32F378xx and STM32F398xx devices.</li> <li>• Removed flag RCC_FLAG_RMV which is write only.</li> <li>• Added RCC_CFGR_xxxx_BITNUMBER definitions to ensure portability between STM32 series.</li> <li>• Updated HAL_RCC_OscConfig() function to enable PWR only if necessary for LSE configuration.</li> </ul> |

| Date        | Revision         | Changes                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 06-Jun-2016 | 4<br>(continued) | <p><b>HAL RTC:</b> added missing Tamper definitions (RTC_TAFCR register).</p> <p><b>HAL SMARTCARD:</b></p> <ul style="list-style-type: none"> <li>• Modified SMARTCARD_Receive_IT() to execute the RX flush request only when no data are read from RDR.</li> <li>• Added SMARTCARD_STOPBITS_0_5 definition used for smartcard mode.</li> <li>• Updated SMARTCARD_SetConfig() function following UART Baudrate calculation update.</li> </ul> <p><b>HAL SPI:</b></p> <ul style="list-style-type: none"> <li>• Updated HAL_SPI_TransmitReceive() function in slave mode to correctly receive the CRC when NSS pulse activated.</li> <li>• Added missing __IO in SPI_HandleTypeDef definition.</li> <li>• Updated IS_SPI_CRC_POLYNOMIAL macro definition as polynomial value should be odd only.</li> </ul> <p><b>HAL TIM:</b></p> <ul style="list-style-type: none"> <li>• Updated HAL_TIM_ConfigOCrefClear() function to correctly manage TIM state (BUSY, READY).</li> <li>• Used IS_TIM_HALL_INTERFACE_INSTANCE macro instead of IS_TIM_XOR_INSTANCE macro in HAL_TIMEx_HallSensor_xxx() functions.</li> <li>• Updated TIM_SLAVE MODE constants definitions using CMSIS bit definitions.</li> </ul> <p><b>HAL UART-USART:</b></p> <ul style="list-style-type: none"> <li>• Updated USART_SetConfig() function following UART Baudrate calculation update.</li> <li>• Removed USART2 and USART3 clock switch, which are not supported by STM32F303x8, STM32F334x8 and STM32F328xx devices and for STM32F301x8, STM32F302x8 and STM32F318xx devices.</li> <li>• Modified UART_Receive_IT() to execute the RX flush request only when no data are read from RDR.</li> <li>• Corrected macro used in assert_param of HAL_LIN_SendBreak() function.</li> <li>• Added USART_STOPBITS_0_5/USART_STOPBITS_0_5 definitions used for synchronous mode.</li> </ul> <p><b>HAL USB:</b> corrected double buffer implementation in PCD_SET_EP_DBUF1_CNT() macro.</p> <p><b>HAL WWDG:</b> aligned WWDG registers bits naming between all STM32 series.</p> |

| Date        | Revision | Changes                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 19-Jul-2016 | 5        | <p>Added Low-Level drivers for ADC, COMP, Cortex, CRC, DAC, DMA, EXTI, GPIO, HRTIM, I2C, IWDG, OPAMP, PWR, RCC, RTC, SPI, TIM, USART and WWDG peripherals and additional Low Level Bus, System and Utilities APIs.</p> <p><b>HAL ADC:</b></p> <ul style="list-style-type: none"> <li>Updated assert_param within HAL_ADCEx_MultiModeConfigChannel() function to avoid issue during ADC configuration change from multimode to independent mode.</li> </ul> <p><b>HAL CRC:</b></p> <ul style="list-style-type: none"> <li>Updated HAL_CRC_DeInit() function (restored IDR Register to Reset value).</li> </ul> <p><b>HAL I2C:</b></p> <ul style="list-style-type: none"> <li>Updated I2C driver description concerning I2C address management.</li> </ul> <p><b>HAL IWDG:</b> New simplified HAL IWDG driver: removed HAL_IWDG_Start(), HAL_IWDG_MspInit() and HAL_IWDG_GetState() APIs. The API functions are:</p> <ul style="list-style-type: none"> <li>HAL_IWDG_Init(): this function insures the configuration and the launch of the IWDG counter.</li> <li>HAL_IWDG_Refresh(): this function insures the reload of the IWDG counter.</li> </ul> <p><b>HAL PWR:</b></p> <ul style="list-style-type: none"> <li>Aligned EWUPx power wakeup pins definitions with CMSIS definitions.</li> </ul> <p><b>HAL SMBUS:</b></p> <ul style="list-style-type: none"> <li>Updated SMBUS address management in SMBUS driver description.</li> </ul> <p><b>HAL SPI:</b></p> <ul style="list-style-type: none"> <li>Updated __SPI_HandleTypeDef definition by using __IO uint16_t type for TxXferCount and RxXferCount.</li> <li>Updated SPI_2linesTxISR_8BIT() and SPI_2linesTxISR_16BIT() functions: added return so that SPI_2linesTxISR_8BITCRC() or SPI_2linesTxISR_16BITCRC() function is called from HAL_SPI_TransmitReceive_IT() when CRC is activated.</li> </ul> <p><b>HAL WWDG:</b></p> <ul style="list-style-type: none"> <li>New simplified HAL WWDG driver: removed HAL_WWDG_Start(), HAL_WWDG_Start_IT(), HAL_WWDG_MspInit() and HAL_WWDG_GetState() APIs.</li> <li>Updated HAL_WWDG_Refresh() API to remove counter parameter.</li> <li>Added new field EWIMode in WWDG_InitTypeDef to specify Early Wakeup Interrupt.</li> <li>The API functions are: HAL_WWDG_Init(), HAL_WWDG_MspInit(), HAL_WWDG_Refresh(), HAL_WWDG_IRQHandler() and HAL_WWDG_EarlyWakeUpCallback().</li> </ul> |
| 02-Jan-2017 | 6        | Update for release V1.4.0 of the HAL and LL drivers. Refer to the release note provided within the firmware package for details.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 10-Jul-2017 | 7        | Update for release V1.5.0 of the HAL and LL drivers. Refer to the release note provided within the firmware package for details.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

**IMPORTANT NOTICE – PLEASE READ CAREFULLY**

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2017 STMicroelectronics – All rights reserved